

普通高等教育“十一五”国家级规划教材  
新编计算机类本科规划教材

# Visual Basic 程序设计教程

## (第4版)

刘瑞新 等编著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

## 内 容 简 介

本书是普通高等教育“十一五”国家级规划教材。

本书以 Visual Basic 6.0 中文版为语言背景,通过大量实例,深入浅出地介绍 Visual Basic 程序开发环境, Visual Basic 程序设计基础,可视化编程的概念与方法,顺序结构程序设计,选择结构程序设计,循环结构程序设计,数组,过程,变量与过程的作用域,用户定义类型与枚举类型,图形与图像,菜单、工具栏与对话框,键盘与鼠标事件过程,数据文件,数据库访问技术等内容。书中每章均附有典型习题。本书免费提供电子课件,可以登录华信教育资源网(<http://www.hxedu.com.cn>),注册后下载。另外,本书有配套的习题解答,对书中习题做了详细解答。

本书可作为大学、高职高专院校的教材使用,本书同样适合作为全国计算机等级考试二级 Visual Basic 语言的辅导教材。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有,侵权必究。

## 图书在版编目(CIP)数据

Visual Basic 程序设计教程 / 刘瑞新等编著. —4 版. —北京: 电子工业出版社, 2011.9

新编计算机类本科规划教材

ISBN 978-7-121-14642-8

I. ①V… II. ①刘… III. ①BASIC 语言—程序设计—高等学校—教材 IV. ①TP312

中国版本图书馆 CIP 数据核字(2011)第 191380 号

策划编辑: 冉 哲

责任编辑: 冉 哲

印 刷:

装 订:

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编 100036

开 本: 787×1092 1/16 印张: 21 字数: 536 千字

印 次: 2011 年 9 月第 1 次印刷

印 数: 4 000 册 定价: 36.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 [zlts@phei.com.cn](mailto:zlts@phei.com.cn), 盗版侵权举报请发邮件至 [dbqq@phei.com.cn](mailto:dbqq@phei.com.cn)。

服务热线: (010) 88258888。

# 前 言

本书是普通高等教育“十一五”国家级规划教材。本书第1版自从2000年1月出版以来，受到广大师生的欢迎，被许多学校常年选为教材。当时，本书是第一本以程序结构为主线，把控件融合到程序结构中讲授的VB教材，形成了可视化类语言教材的编写结构，成为VB教材的范本，从此众多VB教材均采用这种结构组织内容。2003年4月，作者根据教学改革的成功经验并结合读者的建议进行了修订，出版了第2版。第2版出版发行后，以其教学过程自然平顺、学生接受轻松快捷的特点，再次受到读者的青睐。2007年8月，作者根据VB课程教学改革的深入和精品课程建设的需要，对本书进行了第3次修订，第3版秉承了第2版条理清晰、深入浅出、重点突出、难点分散、示例丰富的特点，更加受到读者的欢迎。本书自出版以来已经重印30余次，印数达20余万册，销售量位居同类教材前列。

教学要改革，而教材是影响教学改革的重要因素之一。在教学中，作者不断探索、改革，研究教学方法，并把这些教学改革和创新应用到所编写的教材中，编写了第4版。本书内容均来自教学实践，是对“讲义→教学→修改教学讲义→再次教学→出版教材”整个过程的精确提炼，能够对教师教学、学生学习发挥重要作用。在第4版中，理论与实践结合地恰到好处，具有以下鲜明的特色。

## 1. 可作为程序设计入门教材

学习本书可以没有任何程序设计知识的基础。本书重点讲解计算机语言的基本知识（语言基本元素与结构、语言本身所支持的数据类型、数组、各种表达式的使用），结构化程序设计知识（程序的输入和输出、程序的控制结构、顺序结构、选择结构、循环结构、子程序及文件的使用等），面向对象程序设计的概念与方法，程序中常用的算法等。教材的基本内容主要围绕“程序设计”这个主题。

## 2. 思路清晰、实例典型

本书通过大量有趣的实例介绍程序设计基础、方法，避免枯燥、空洞的理论，容易上手，使读者于不知不觉之中学会在Windows环境下编程。本书在讲解例题时，首先给出设计目标，然后介绍为实现本目标而采取的设计方法。采用这种处理方式，可使学生明确程序设计的思想和方法，做到有的放矢。

## 3. 分析透彻、重点突出

本书以Visual Basic中文版为语言背景，通过大量实例，深入浅出地介绍了Visual Basic程序开发环境，Visual Basic程序设计基础，可视化编程的概念与方法，顺序结构程序设计，选择结构程序设计，循环结构程序设计，数组，过程，变量与过程的作用域，用户定义类型与枚举类型，图形与图像，菜单、工具栏与对话框，键盘与鼠标事件过程，数据文件，数据库访问技术等内容。本书采用Visual Basic作为编程环境，所有程序都可以在Visual Basic 6.0下正常运行。

#### 4. 循序渐进、层次分明

针对初学者的特点，全书在编排上注意由简到繁、由浅入深和循序渐进的特点，力求通俗易懂、简单实用。本书概念清晰、逻辑性强、层次分明、例题丰富，符合教师教学和学生学习习惯。同时，书中的文字在描述上更加准确精练，以浅显易懂的语言和具有代表性的示例，使“复杂的问题简单化、重要的问题深刻化”。

#### 5. 符合全国计算机等级考试大纲要求

本书涵盖了“全国计算机等级考试二级考试大纲（Visual Basic 程序设计）”的内容，增加了大量全国等级考试的试题，学习本书后，可参加全国计算机等级考试。

#### 6. 习题丰富、有配套的习题解答

本书在介绍学生应掌握知识的基础上，以强化学生实践能力为目的，涵盖选择题、填空题、思考题、编程题等各种题型，既能使学生便于检测知识掌握程度，又符合各类 VB 考试题型。另外加大了练习题量，便于教师进行题库建设。

为方便学生练习和知识检测，本书有配套的习题解答，对书中习题做了详细解答，还增加了大量全国计算机等级考试二级 Visual Basic 试题和解答。配套使用将使学习效果更佳。

本书免费提供电子课件，可以登录华信教育资源网（<http://www.hxedu.com.cn>）注册后免费下载。

本书由刘瑞新编著，参加编写的作者还有蔡峰、张鸣、刘慧敏、王瑶、胡楠、张志强、张明增、贾俊亮、马志刚、冯全民、董福新、刘美想、张锐、张小兵、杨桦。随着社会发展和教学改革的深入，请读者将教改的成果经验和对本书的建议及时告知，以便精益求精。

本书可作为大学、高职高专院校教材使用，希望广大教师、学生一如既往地支持我们，多提宝贵意见，以便更加完善本书的教学体系。

作 者

# 目 录

第 1 章 Visual Basic 程序开发环境	1
1.1 VB 的特点和版本	1
1.1.1 VB 的特点	1
1.1.2 VB 的版本	2
1.2 VB 的启动与退出	3
1.3 VB 集成开发环境	4
1.3.1 主窗口	4
1.3.2 其他窗口	5
1.3.3 单文档界面和多文档界面	9
1.4 VB 帮助系统	9
习题 1	10
第 2 章 Visual Basic 程序设计基础	13
2.1 基本数据类型	13
2.2 变量	17
2.3 常量	19
2.4 运算符和表达式	20
2.4.1 算术运算符和算术表达式	21
2.4.2 字符串运算符和字符串表达式	22
2.4.3 日期运算符和日期表达式	23
2.4.4 关系运算符和关系表达式	23
2.4.5 逻辑运算符和逻辑表达式	24
2.4.6 表达式的执行顺序	26
2.5 常用内部函数	27
2.5.1 数学运算函数	27
2.5.2 字符串函数	28
2.5.3 日期和时间函数	30
2.5.4 格式输出函数	31
2.5.5 随机数语句和函数	32
2.5.6 数据类型转换函数	33
2.5.7 Shell 函数	34
2.6 语句	35
习题 2	36

第 3 章 Visual Basic 可视化编程的概念与方法 .....	39
3.1 可视化编程的基本概念 .....	39
3.1.1 对象 .....	39
3.1.2 对象的属性、事件和方法 .....	39
3.2 窗体、控件和代码窗口 .....	41
3.2.1 窗体对象 .....	41
3.2.2 控件 .....	41
3.2.3 代码窗口 .....	43
3.3 可视化编程的一般步骤 .....	44
习题 3 .....	48
第 4 章 顺序结构程序设计 .....	49
4.1 顺序结构程序的概念 .....	49
4.2 数据输出 .....	49
4.2.1 直接输出到窗体 .....	49
4.2.2 使用标签控件输出 .....	55
4.3 常用基本语句 .....	56
4.3.1 赋值语句 Let .....	56
4.3.2 卸载对象语句 Unload .....	58
4.3.3 注释语句 Rem .....	58
4.4 利用文本框输入数据 .....	59
4.4.1 文本框控件 .....	59
4.4.2 焦点与 Tab 键序 .....	62
4.4.3 框架控件 .....	63
4.5 使用对话框 .....	64
4.5.1 输入框 (InputBox) 函数 .....	65
4.5.2 消息框 (MsgBox) 函数 .....	66
习题 4 .....	68
第 5 章 选择结构程序设计 .....	71
5.1 If 语句 .....	71
5.1.1 单行结构条件语句 If...Then...Else .....	71
5.1.2 块结构条件语句 If...Then...Else...End If .....	73
5.1.3 使用 IIf 函数 .....	74
5.1.4 If 语句的嵌套 .....	75
5.2 多分支条件选择语句 Select Case .....	78
5.3 计时器控件 .....	83
5.4 单选钮和复选框 .....	86
5.4.1 单选钮控件 .....	86
5.4.2 复选框控件 .....	91

习题 5	92
第 6 章 循环结构程序设计	96
6.1 For...Next 循环语句	96
6.2 Do...Loop 循环语句	100
6.2.1 前测型 Do...Loop 循环语句	100
6.2.2 后测型 Do...Loop 循环语句	104
6.3 列表框与组合框	107
6.3.1 列表框控件	107
6.3.2 组合框控件	112
习题 6	115
第 7 章 数组	122
7.1 数组和数组元素	122
7.2 静态数组	124
7.2.1 声明静态数组	124
7.2.2 Option Base 语句	124
7.2.3 数组的基本操作	125
7.2.4 数组元素的输入、输出和复制	125
7.2.5 数组的初始化	127
7.2.6 静态数组使用示例	127
7.3 动态数组	134
7.3.1 创建动态数组	134
7.3.2 保留动态数组的原有数据	135
7.4 For Each...Next 语句	136
7.5 控件数组	138
7.5.1 控件数组的概念	138
7.5.2 控件数组的建立	139
7.5.3 控件数组使用示例	140
习题 7	145
第 8 章 过程	150
8.1 事件过程	150
8.2 子过程	151
8.2.1 创建子过程	152
8.2.2 调用子过程	153
8.2.3 子过程使用示例	153
8.3 函数过程	157
8.3.1 定义函数过程	157
8.3.2 调用函数过程	158

8.3.3	函数过程使用示例	159
8.3.4	查看过程	161
8.4	参数传递	161
8.4.1	形式参数与实际参数	161
8.4.2	按值传递与按地址传递	162
8.4.3	使用参数	164
8.4.4	传递数组	166
8.5	过程的嵌套与递归调用	169
8.5.1	过程的嵌套调用	169
8.5.2	过程的递归调用	170
习题 8		173
第 9 章	变量与过程的作用域	178
9.1	代码模块的概念	178
9.2	变量的作用域和生存期	180
9.2.1	变量的作用域	180
9.2.2	变量的生存期	182
9.3	过程的作用域	185
9.4	按钮控件	186
习题 9		187
第 10 章	用户定义类型与枚举类型	190
10.1	用户定义类型	190
10.1.1	建立用户定义类型	190
10.1.2	建立和使用用户定义类型变量	191
10.1.3	用户定义类型数组	192
10.2	枚举类型	194
10.2.1	定义枚举类型	194
10.2.2	枚举类型使用示例	195
10.3	滚动条控件	197
10.3.1	滚动条控件的类型	197
10.3.2	滚动条控件的常用属性	197
10.3.3	滚动条控件的常用事件	198
10.3.4	滚动条控件使用示例	198
习题 10		201
第 11 章	图形与图像	203
11.1	绘制图形	203
11.1.1	图形控件	203
11.1.2	图形的坐标系统	206



11.1.3	与图形有关的属性	207
11.1.4	使用颜色	210
11.1.5	常用绘图方法	211
11.1.6	绘图语句与 Paint 事件	216
11.2	显示图片	216
11.2.1	直接加载图片到窗体	217
11.2.2	使用图像控件	217
11.2.3	使用图片框控件	219
习题 11		223
第 12 章	菜单、工具栏与对话框	226
12.1	菜单	226
12.1.1	菜单的两种基本类型	226
12.1.2	菜单编辑器	227
12.1.3	设计下拉式菜单	228
12.1.4	设计弹出式菜单	234
12.2	工具栏	237
12.2.1	手工方式设计工具栏	237
12.2.2	使用工具栏控件设计工具栏	238
12.3	公共对话框	242
12.3.1	添加公共对话框控件	242
12.3.2	使用公共对话框控件	242
12.3.3	公共对话框控件的应用举例	246
习题 12		247
第 13 章	键盘与鼠标事件过程	253
13.1	键盘事件	253
13.1.1	KeyPress 事件	253
13.1.2	KeyDown 事件和KeyUp 事件	254
13.1.3	使用 KeyPreview 属性	256
13.2	鼠标事件	256
13.2.1	MouseDown 事件和 MouseUp 事件	257
13.2.2	MouseMove 事件	257
13.2.3	自定义鼠标指针	257
13.2.4	使用鼠标事件	259
13.3	拖放事件	261
13.3.1	与拖放有关的属性、事件与方法	261
13.3.2	自动拖放	263
13.3.3	手工拖放	265
习题 13		266

第 14 章 数据文件	270
14.1 文件的分类与结构	270
14.1.1 文件的分类	270
14.1.2 文件的结构	271
14.2 文件操作语句和函数	271
14.2.1 数据文件的操作	271
14.2.2 文件的打开与关闭语句	272
14.2.3 文件访问函数	273
14.3 顺序文件的操作	275
14.3.1 顺序文件的写操作	275
14.3.2 顺序文件的读操作	278
14.4 随机文件的操作	284
14.4.1 随机文件的读/写操作	284
14.4.2 随机文件中记录的增加与删除	287
14.5 文件系统控件	288
14.5.1 驱动器列表框	288
14.5.2 目录列表框	289
14.5.3 文件列表框	290
14.5.4 文件系统控件共有的属性	290
14.5.5 文件系统对象的同步操作	290
14.6 文件基本操作	291
14.6.1 目录的基本操作	291
14.6.2 文件的基本操作	292
习题 14	293
第 15 章 数据库访问技术	297
15.1 数据库的概念	297
15.2 Access 数据库	298
15.2.1 创建 Access 数据库和表	298
15.2.2 创建查询	299
15.3 使用数据控件	301
15.3.1 数据控件的属性	301
15.3.2 数据控件的事件	303
15.3.3 数据控件的方法	303
15.3.4 记录集对象	304
15.4 使用 ADO 控件	312
15.4.1 ADO 数据控件的属性、方法和事件	312
15.4.2 高级数据绑定控件	315
15.4.3 使用数据窗体向导	320
习题 15	322

# 第 1 章 Visual Basic 程序开发环境

Visual Basic（简称 VB）是美国 Microsoft 公司推出的 Windows 环境下的软件开发工具，使用 VB 可以轻松快捷地开发 Windows 应用软件。

## 1.1 VB 的特点和版本

BASIC（Beginners All-purpose Symbolic Instruction Code，初学者通用符号指令代码）语言于 1964 年问世，是国际上广泛使用的一种计算机高级语言。1991 年，微软推出 Visual Basic 1.0 版，在当时引起了很大的轰动。许多专家把 Visual Basic 的出现当做软件开发史上的一个具有划时代意义的事件。Visual Basic（简称 VB）是从 BASIC 语言发展而来的，是在 Windows 环境下快速开发应用程序的可视化工具。其中，Visual 是指开发图形用户界面（GUI）的方法。Visual 的英文原意是“视觉的”或“可视的”，这里是指直观的编程方法。之所以叫做 Visual Basic，是因为它使用了 BASIC 语言作为编程语言。

### 1.1.1 VB 的特点

VB 是目前所有开发语言中最简单、最容易使用的语言。作为程序设计语言，VB 主要有以下特点。

#### 1. 面向对象的可视化设计平台

利用传统的程序设计语言进行程序设计时，需要花费大量的精力编程设计程序的界面，在设计过程中看不到程序的实际显示效果，必须在运行程序的时候才能看到。如果对程序的界面不满意，还要回到程序中去修改，这一过程常常需要反复多次，大大影响了编程的效率。

VB 提供的可视化设计平台，把 Windows 界面设计的复杂性“封装”起来，利用控件（如各种各样的按钮、文本框、复选框等）并将这些控件进行模式化，而且每个控件都由若干属性来控制其外观、工作方法。这样，程序员不必再为界面的设计而编写大量程序代码，只需按设计的要求，把预先建立的控件加到屏幕上，VB 将自动产生界面设计代码，程序员所需要编写的只是实现程序功能的那部分代码，从而大大提高了编程的效率。

#### 2. 事件驱动的编程机制

传统的编程方式是面向过程的、按事先设计的程序流程来运行的。但在图形用户界面的应用程序中，用户的动作（即事件）控制着程序的运行流向，每个事件都驱动一段程序的运行。程序员在设计应用程序时，不必建立具有明显开始和结束的程序，而是编写若干个微小的子程序，即过程。这些过程分别面向不同的对象，由用户操作引发某个事件来驱动完成某种特定的功能，或者由事件驱动程序调用通用过程来执行指定的操作。

### 3. 结构化的设计语言

VB 是在结构化的 BASIC 语言基础上发展起来的，具有高级程序设计语言的语句结构，接近于自然语言和人类的逻辑思维方式，其语句简单易懂；其编辑器可自动进行语法错误检查，同时具有功能强且使用灵活的调试器和编译器。

VB 是解释型语言，在输入代码的同时，解释系统将高级语言分解翻译成计算机可以识别的机器指令，并判断每个语句的语法错误。在设计 VB 程序的过程中，随时可运行程序，而在整个应用程序设计好后，可编译生成可执行文件(.exe)，脱离 VB 环境，直接在 Windows 环境下运行。

### 4. 充分利用 Windows 资源

VB 提供的动态数据交换 (DDE, Dynamic Data Exchange) 编程技术，可以在应用程序中实现与其他 Windows 应用程序建立动态数据交换、在不同的应用程序之间进行通信的功能。

VB 提供的对象链接与嵌入 (OLE, Object Link and Embed) 技术则将每个应用程序都看做一个对象，将不同的对象链接起来，嵌入到某个应用程序中，从而得到具有声音、影像、图像、动画、文字等各种信息的集合式文件。

VB 还可以通过动态链接库 (DLL, Dynamic Link Library) 技术将 C/C++ 或汇编语言编写的程序加入到 VB 的应用程序中，或者调用 Windows 应用程序接口 (API, Application Programming Interfaces) 函数，实现软件开发工具包 (SDK, Software Development Kit) 所具有的功能。

### 5. 开放的数据库功能与网络支持

VB 系统具有很强的数据库管理功能。不仅可以管理 MS Access 格式的数据库，还能访问其他外部数据库，如 FoxPro, Paradox 等格式的数据库。另外，VB 还提供了开放式数据连接 (ODBC) 功能，可以通过直接访问或建立连接的方式使用并操作后台大型网络数据库，如 SQL Server, Oracle 等。在应用程序中，可以使用结构化查询语言 (SQL) 直接访问 Server 上的数据库，并提供简单的面向对象的库操作命令、多用户数据库的加锁机制和网络数据库的编程技术，为单机上运行的数据库提供 SQL 网络接口，以便在分布式环境中快速而有效地实现客户-服务器 (Client/Server) 方案。

#### 1.1.2 VB 的版本

1991 年，微软公司发布第一个 Visual Basic 版本——Visual Basic 1.0 版。1992 年，经过对 Visual Basic 1.0 的修改后，发布 Visual Basic 2.0 版。1993 年，经再次修改完善后，发布 Visual Basic 3.0 版。1995 年，发布 Visual Basic 4.0 版。1997 年，微软公司发布最早的 Windows 开发工具套件 Visual Studio 97，其中包含 Visual Basic 5.0 版。1998 年，发布 Visual Studio 6.0 (也称 Visual Studio 98)，其中包含 Visual Basic 6.0 版。2002 年，随着 .NET 的提出与 Windows XP/Office XP 的发布，微软发布 Visual Studio .NET，其中包含 Visual Basic .NET 2002 (v7.0) 版。2003 年，发布 Visual Studio 2003，其中包含 Visual Basic .NET 2003 (v7.1)。2005 年，发布 Visual Studio 2005，其中包含 Visual Basic 2005 (v8.0)。2008 年，发布 Visual Studio 2008，其中包含 Visual Basic 2008 (v9.0)。2010 年，发布 Visual Studio 2010，其中包含 Visual

Basic 2010 (v10.0)。

因为 Visual Basic 6.0 操作简单，所以现在仍然作为入门的语言环境来学习。Visual Basic 6.0 包括三种版本：学习版、专业版和企业版。三种版本适合不同的用户层次。这些版本是在相同的基础上建立起来的，因此大多数应用程序可在三种版本中通用。

- 学习版：是 Visual Basic 的基础版本，可用来开发 Windows 应用程序。该版本包括了所有的内部控件（标准控件）、网格（Grid）控件、Tab 对象及数据绑定控件。
- 专业版：该版本为专业编程人员提供了一整套用于软件开发、功能完备的工具。它包括了学习版的全部功能，同时包括 ActiveX 控件、Internet 控件和报表控件等。
- 企业版：可供专业编程人员使用的，功能强大的组内分布式应用程序。该版本包括了专业版的全部功能，还增加了自动化管理器、部件管理器、数据库管理工具、Microsoft Visual SourceSafe 面向工程版的控制系统等工具。

本书选用 Visual Basic 6.0 企业版作为学习环境，但书中程序仍然可在专业版和学习版中运行。


Visual Basic 6.0 是专门为 Windows 9x/NT/2000 等 32 位操作系统设计的。用 Visual Basic 6.0 的编译器可以自动生成 32 位应用程序，可脱离 Visual Basic 6.0 的运行环境，且运行速度更快、更安全，适合在多任务环境下运行。

## 1.2 VB 的启动与退出

Visual Basic 6.0 可以运行在 Windows 95/98/Me/NT/2000/XP 下，为了叙述方便，在本书中统称为 Windows。此外，除特别说明外，Visual Basic 6.0 简称为 VB。

### 1. VB 的启动

启动 VB 的步骤如下。

(1) 单击 Windows 任务栏中的“开始”按钮  → “程序” → “Microsoft Visual Basic 6.0 中文版” → “Microsoft Visual Basic 6.0 中文版”项，即可启动 VB。

(2) 启动 VB 后，首先显示“新建工程”对话框，如图 1-1 所示。




图 1-1 “新建工程”对话框

(3) 系统默认为“新建”选项卡中的“标准 EXE”项。双击“标准 EXE”项，或者直接单击“打开”按钮，进入 VB 的集成开发环境。

在集成开发环境中集中了许多不同的功能，如程序设计、编辑、编译和调试等。

## 2. VB 的退出

如果要退出 VB，可单击 VB 窗口右上角的“关闭”按钮 ，或者选择“文件”→“退出”菜单命令，VB 会自动判断用户是否修改了工程的内容，并询问用户是否保存文件或直接退出。

# 1.3 VB 集成开发环境

VB 把支持软件开发的各功能都集成在一个公共的工作环境中，称为“集成开发环境”，如图 1-2 所示。在集成开发环境中，集中提供了程序开发所需要的各种工具、窗口和方法，用户可以方便地在软件开发的各阶段工作中来回切换，从而提高开发效率。

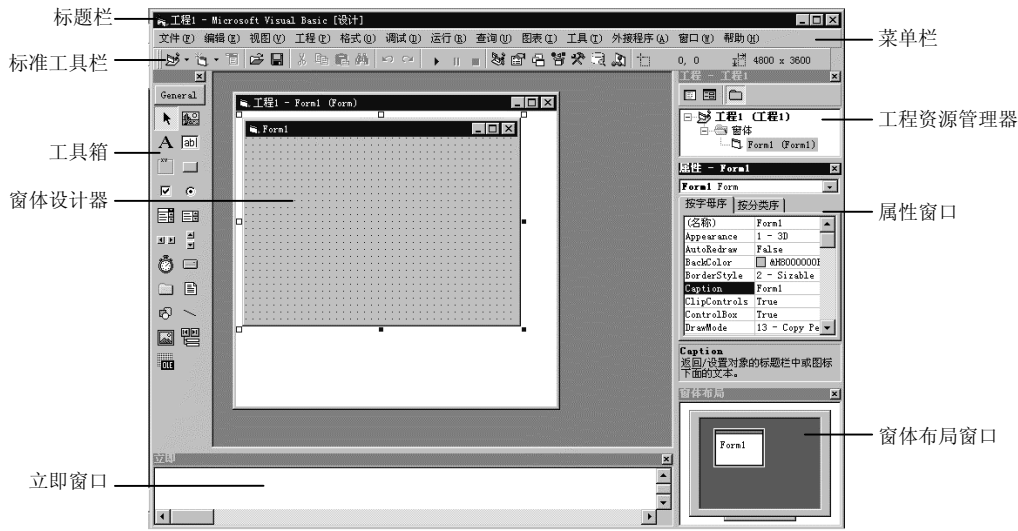


图 1-2 VB 集成开发环境

## 1.3.1 主窗口

### 1. 标题栏

标题栏中显示的有窗体控制菜单图标 、当前激活的工程名称、当前工作模式，以及最小化按钮、最大化/还原按钮、关闭按钮。

### 2. 菜单栏

菜单栏中显示了“文件”、“编辑”、“视图”、“工程”、“格式”等菜单项，其中包含了 VB 编程的常用命令。单击菜单栏中的菜单名，即可弹出下拉菜单。在下拉菜单中显示各种功能子菜单，包含执行该项功能的热键和快捷键。

3. 工具栏

VB 提供了 4 种工具栏，包括标准工具栏、编辑工具栏、窗体编辑器工具栏和调试工具栏，用户还可以根据需要定义自己的工具栏。在一般情况下，集成环境中只显示标准工具栏，其他工具栏可以通过“视图”→“工具栏”菜单命令打开或关闭。

标准工具栏中提供了许多常用命令的快速访问按钮，单击某个按钮，即可执行相应的菜单操作。标准工具栏中各按钮的图标及其功能见表 1-1。

表 1-1 标准工具栏中各按钮的图标及其功能

图 标	名 称	功 能	快 捷 键
	添加标准 EXE 工程	用来添加新的工程到工作组中，单击其右边的下拉箭头，将弹出一个下拉菜单，可以从中选择需要添加的工程类型	
	添加窗体	用来添加新的窗体到工程中，单击其右边的下拉箭头，将弹出一个下拉菜单，可以从中选择需要添加的窗体类型	
	菜单编辑器	显示菜单编辑器	Ctrl+E
	打开工程	用于打开已有的工程文件	Ctrl+O
	保存工程	用于保存当前的工程文件	
	启动	开始运行当前的工程	F5
	中断	暂时中断当前工程的运行	Ctrl+Break
	结束	结束当前工程的运行	
	工程资源管理器	打开工程资源管理器	Ctrl+R
	属性窗口	打开属性窗口	F4
	窗体布局窗口	打开窗口布局窗口	
	对象浏览器	打开对象浏览器	F2
	工具箱	打开工具箱	
	数据视图窗口	打开数据视图窗口	
	可视化部件管理器	打开可视化部件管理器	

在标准工具栏右侧还有两个栏，分别用来显示窗体的当前位置和大小，其单位是 twip（1 英寸=1 440twip）。左边一栏显示对象的左上角的坐标，窗体设计器的左上角为坐标原点，即（0,0）位置；右边一栏显示的是对象的高度（向下递增）和宽度，即对象的大小。

twip 是一种与屏幕分辨率无关的计量单位，无论在什么显示方式下，如果画一条 1 440twip 的直线，打印出来都是 1 英寸长。这种计算单位可以确保在不同的屏幕上保持正确的相对位置或比例关系。在 VB 中，twip 是默认单位，可以通过 ScaleMode 属性改变。

1.3.2 其他窗口

标题栏、菜单栏和工具栏所在的窗口称为主窗口。除主窗口外，VB 的编程环境中还有一些其他窗口，包括窗体设计器、工程资源管理器、属性窗口、工具箱、窗体布局窗口和立即窗口。

1. 窗体设计器

窗体设计器简称窗体（Form），就是应用程序最终面向用户的窗口，它对应于应用程序

的运行结果。各种图形、图像、数据等都是通过窗体或窗体中的控件显示出来的。当创建一个新的工程文件时，VB 建立一个空的窗体。

每次启动 VB 后，窗体的默认名称为 Form1。在其操作区中布满了排列整齐的小点，如图 1-3（a）所示，这些小点是供对齐窗体中的对象用的。如果要清除小点或改变点与点之间的距离，可通过执行“工具”→“选项”菜单命令，在打开的“选项”对话框的“通用”选项卡中调整。

窗体的左上角显示的是窗体的标题，右上角有三个按钮，其作用与 Windows 普通窗口中的作用相同，即最小化、最大化、关闭按钮。

在设计应用程序时，窗体就像一块画布，在这块画布上可以“画”出组成应用程序的各个“构件”，如图 1-3（b）所示。程序员根据程序界面的要求，从工具箱中选取需要的“构件”（称为控件），在窗体中画出来，这样就完成了应用程序设计的第一步。

一般，窗体中的控件可在窗体上随意移动、改变大小，锁定后则不可随意修改。

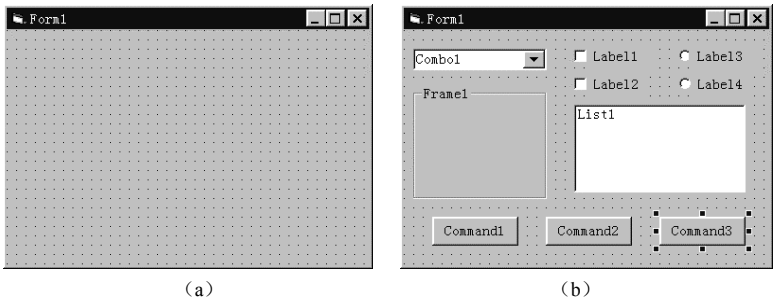


图 1-3 窗体

2. 工程资源管理器

工程是指用于创建一个应用程序的所有文件的集合。工程资源管理器（简称工程窗口）采用 Windows 资源管理器式的界面，层次分明地列出当前工程中所有文件的清单，如图 1-4 所示。工程窗口中的文件可以分为 6 类：工程文件（.vbp）、工程组文件（.vbg）、窗体文件（.frm）、程序模块文件（.bas）、类模块文件（.cls）和资源文件（.res）。



图 1-4 工程资源管理器

（1）工程文件和工程组文件

工程文件就是与该工程有关的所有文件和对象的清单，这些文件和对象自动链接到工程文件上，每次保存工程时，其相关文件信息随之更新。当然，某个工程下的对象和文件也可供其他工程共享使用。在工程的所有对象和文件被汇集在一起并完成编码以后，就可以编译工程，生成可执行文件了。

工程文件的扩展名为.vbp，每个工程对应一个工程文件。当一个程序包括两个以上的工程时，这些工程构成一个工程组，工程组文件的扩展名为.vbg。执行“文件”→“新建工程”



菜单命令可以建立一个新的工程，执行“打开工程”命令可以打开一个已有的工程，而执行“添加工程”命令可以添加一个工程。

## （2）窗体文件

窗体文件的扩展名为.frm，每个窗体对应一个窗体文件，窗体及其控件的属性和其他信息（包括代码）都存放在该窗体文件中。一个应用程序可以有多个窗体（最多 255 个），因此就可以有多个以.frm 为扩展名的窗体文件。

执行“工程”→“添加窗体”菜单命令，或单击工具栏中的“添加窗体”按钮，可以增加一个窗体；而执行“工程”→“移除窗体”菜单命令，可以删除当前的窗体。每建立一个窗体，工程窗口中就增加一个窗体文件，每个窗体都有一个不同的名字，可以通过属性窗口设计（Name 属性），其默认名字依次为 Form1, Form2, Form3 等。

## （3）程序模块文件

程序模块文件的扩展名是.bas，它是为合理组织程序而设计的。程序模块是一个纯代码性质的文件，它不属于任何一个窗口，主要在大型应用程序中使用。




程序模块文件由程序代码组成，主要用来声明全局变量和定义一些通用的过程，可以被多个不同窗体中的程序调用。程序模块通过“工程”→“添加模块”菜单命令来建立。

## （4）类模块文件

VB 中提供了大量预定义的类，同时也允许用户根据需要定义自己的类。用户通过类模块来定义自己的类，每个类都用一个文件来保存，其扩展名为.cls。

## （5）资源文件

资源文件是一种可以同时存放文本、图片、声音等多种资源的文件。它由一系列独立的字符串、位图及声音文件（.wav, .mid）组成，其扩展名为.res。资源文件是一种纯文本文件，可以用文字编辑器编辑。

在工程窗口的顶部还有三个按钮，分别是：“查看代码”、“查看对象”和“切换文件夹”。单击“查看代码”按钮，可打开代码编辑器查看代码；单击“查看对象”按钮，可打开窗体设计器查看正在设计的窗体；单击“切换文件夹”按钮，则可以隐藏或显示包含对象文件夹中的个别项目列表。

在设计应用程序时，通常先设计窗体（即界面），然后再编写程序。窗体设计完成后，只要双击窗体中的对象，就可以切换到代码编辑器窗口中，与单击“查看代码”按钮的作用相同。

# 3. 属性窗口

在 VB 集成环境的默认视图中，属性窗口位于工程窗口的下面。按 F4 键，或者单击工具栏中的“属性窗口”按钮，或者执行“视图”→“属性窗口”菜单命令，均可打开属性窗口，如图 1-5 所示。

属性窗口主要是针对窗体和控件设置的。在 VB 中，窗体和控件被称为对象。每个对象都可以用一组属性来描述其特征，而属性窗口就是用来设置窗体或窗体中控件的属性的。

属性窗口中包含选定对象（窗体或控件）的属性列表，在设计程序时可通过修改对象的属性来改变其外观和相关数据，这些属性值将是程序运行时各对象属性的初始值。属性窗口分为 4 部分。



图 1-5 属性窗口

① 对象下拉列表框：显示当前选定对象的名称及所属的类。单击右端的下拉箭头，可列出当前窗体及所包含的全部对象的名称，可从中选择要更改其属性的对象。

② 选项卡：有按字母排序或按分类排序两种方式显示所选对象的属性。

③ 属性列表框：属性列表框中列出了当前选定的窗体或控件的属性设置值。左半边显示所选对象的所有属性名，右半边显示属性值。可以直接在属性窗口中修改属性值。有的属性取值具有预定值，如果其右侧显示“三点”式按钮 或“下拉箭头”式按钮 ，则说明有预定值可供选择。在属性列表框中双击属性值可以遍历所有选项。选择任一属性并按 F1 键可得到该属性的帮助信息。

④ 属性说明：显示当前属性的简要说明。可通过右键快捷菜单中的“描述”命令来切换显示或隐藏属性说明。

每个 VB 对象都有其特定的属性，可以通过属性窗口来设置，对象的外观和对应的操作由所设置的值来确定。有些属性的取值是有一定限制的，例如，只能设置为 True 或 False。在实际的应用程序设计中，没有必要设置对象的所有属性，很多属性可以使用默认值。

#### 4. 工具箱

工具箱由工具图标组成，这些图标是 VB 应用程序的构件，称为图形对象或控件，每个控件由工具箱中的一个图标来表示。工具箱主要用于应用程序的界面设计。在设计阶段，首先用工具箱中的工具（控件）在窗体上建立用户界面，然后编写程序代码。界面的设计完全通过控件来实现，可以任意改变其大小，或在窗体中移动到任何位置。

工具箱中的工具分为两类：一类称为内部控件或标准控件，另一类称为 ActiveX 控件。新建或打开“标准 EXE”工程时，将同时打开标准工具箱。

标准工具箱中包含了建立应用程序所需的各种控件，如图 1-6（a）所示。另外，VB 还提供了很多 ActiveX 控件可以添加到工具箱中，图 1-6（b）所示为扩充工具箱。

#### 5. 窗体布局窗口

窗体布局窗口中有一个表示屏幕的小图像，用来显示窗体在屏幕中的位置。可以用鼠标拖动其中的窗体小图标来调整窗体位置。

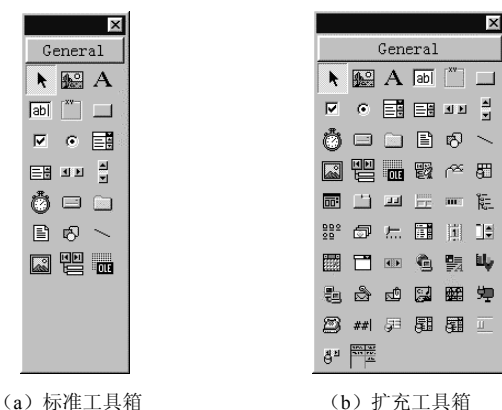


图 1-6 VB 的工具箱

## 6. 立即窗口

使用立即窗口可以在中断状态下查询对象的值，也可以在设计时查询表达式的值或命令的结果。立即窗口如图 1-7 所示，前 3 行是输入的命令，第 4 行是输出的结果。



图 1-7 在立即窗口中输出表达式的值

## 1.3.3 单文档界面和多文档界面

VB 应用程序界面有两种不同的类型：单文档界面（SDI）和多文档界面（MDI）。对于单文档界面，所有窗口可在屏幕上任何地方自由移动，但是同一时间内只能打开一个窗口，并且只要 VB 是当前应用程序，它们将位于其他应用程序之上。对于多文档界面（MDI），有一个包含多个子窗口的、大小可调的父窗口，可以同时打开多个窗口。

## 1.4 VB 帮助系统

在使用 VB 进行程序设计时，经常会遇到一些问题，特别是对初学者更是如此。MSDN Library 为用户提供了包括 VB 在内的，近 1GB 的编程技术信息和大量的示例程序代码。当用户遇到问题时可以随时查阅。所以掌握和使用好 VB 的帮助系统是十分重要的。

### 1. 使用 MSDN Library 在线帮助

在“帮助”菜单中选择“内容”、“索引”或“搜索”命令后，将打开类似于 IE 浏览器的 MSDN Library 在线帮助窗口，如图 1-8 所示。

该窗口中包含定位和主题两个窗格。在定位窗格中有“目录”、“索引”、“搜索”和“书签”4 个选项卡，选择了某个选项卡后，即可在主题窗格中查看有关的信息。例如，选择“搜索”选项卡后可以输入单词或短语，以便快速获得需要的帮助信息。



图 1-8 MSDN Library Visual Studio 6.0 窗口

在主题窗格中，有些文字带下划线（超链接文字），单击这些文字可以获得进一步的解释和说明或链接到其他主题和网页。

## 2. 上下文相关帮助

VB 的许多部分是上下文相关的。上下文相关意味着不必搜寻“帮助”菜单就可直接获得有关帮助。例如，选中窗体，如图 1-9 所示，按 F1 键，将显示图 1-10 所示的帮助信息。

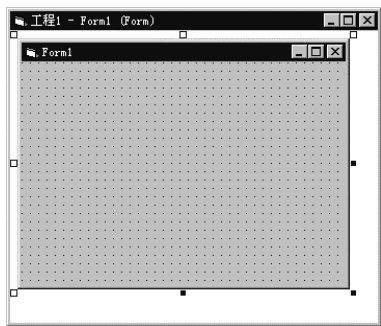


图 1-9 选中窗体



图 1-10 按 F1 键取得帮助

## 3. 运行“帮助”中的代码示例

为了帮助对概念的理解，VB 帮助系统中包含有一些可以在 VB 中直接运行的代码示例，可以通过 Windows 的剪贴板将这些代码复制到代码窗口中，并按 F5 键运行。但有些需要先建立窗体和控件，设置属性后才能运行示例代码。

## 习题 1

### 一、选择题

1.1 Visual Basic 6.0 分为 3 种版本，不属于这 3 种版本的是（ ）。

- A) 学习版      B) 专业版      C) 企业版      D) 业余版

1.2 下列方法中不能退出 Visual Basic 的是 ( )。

- A) 按 Alt+Q 组合键
- B) 按 Alt+F 组合键, 然后按 Esc 键
- C) 按 F10 键, 然后按 F 键, 再按 X 键
- D) 执行“文件”→“退出”菜单命令

1.3 以下叙述中错误的是 ( )。

- A) Visual Basic 是事件驱动型可视化编程工具
- B) Visual Basic 应用程序不具有明显的开始和结束语句
- C) Visual Basic 工具箱中的所有控件都具有宽度 (Width) 和高度 (Height) 属性
- D) Visual Basic 中控件的某些属性只能在运行时设置

1.4 以下叙述中错误的是 ( )。

- A) 在工程资源管理器窗口中只能包含一个工程文件及属于该工程的其他文件
- B) 以 .bas 为扩展名的文件是标准模块文件
- C) 窗体文件包含该窗体及其控件的属性
- D) 一个工程中可以含有多个标准模块文件

1.5 以下不属于 Visual Basic 系统的文件类型是 ( )。

- A) frm
- B) bat
- C) vbg
- D) vbp

1.6 以下叙述中错误的是 ( )。

- A) 打开一个工程文件时, 系统自动装入与该工程有关的窗体、标准模块等文件
- B) 保存 Visual Basic 程序时, 应分别保存窗体文件及工程文件
- C) Visual Basic 应用程序只能以解释方式执行
- D) 事件可以由用户引发, 也可以由系统引发

1.7 Visual Basic 集成的主窗口中不包括 ( )。

- A) 属性窗口
- B) 标题栏
- C) 菜单栏
- D) 工具栏

1.8 下列操作可以打开立即窗口的是 ( )。

- A) Ctrl+D
- B) Ctrl+F
- C) Ctrl+G
- D) Ctrl+E

1.9 下列说法错误的是 ( )。

- A) 方法是对象的一部分
- B) 在调用方法时, 对象名是不可缺少的
- C) 方法是一种特殊的过程和函数
- D) 方法调用格式和对象属性使用格式相同

1.10 下列说法错误的是 ( )。

- A) 窗体文件的扩展名为 .frm
- B) 一个窗体对应一个窗体文件
- C) VB 中一个工程只包含一个窗体
- D) VB 中一个工程最多可以包含 255 个窗体

1.11 一个工程必须包含的文件的类型是 ( )。

- A) .vbp ,.frm ,.frx
- B) .vbp ,.cls ,.bas
- C) .bas ,.ocx ,.res
- D) .frm ,.cls ,.bas

1.12 新建一个窗体, 其 BorderStyle 属性设置为 Fixed Single, 但运行时却没有最大化和最小化按钮, 可能的原因是 ( )。

- A) BorderStyle 的值设为 Fixed.Single, 此项设置值的作用是禁止最大化和最小化按钮
- B) 窗体的 MaxButton 和 MinButton 值设为 False

- C) 在正常情况下新建的窗体都没有最大化和最小化按钮
- D) 该窗体可用鼠标拖动框的方法改变窗体的大小

## 二、填空题

- 1.13 与传统的程序设计语言相比，Visual Basic 最突出的特点是\_\_\_\_\_。
- 1.14 如果不使用鼠标，用键盘打开菜单和执行菜单命令，第一步应按\_\_\_\_\_键。
- 1.15 建立一个新的标准模块，应该选择\_\_\_\_\_菜单下的“添加模块”命令。

## 三、思考题

- 1.16 叙述 Visual Basic 6.0 的安装过程。
- 1.17 叙述 MSDN 的安装方法、作用及使用方法。

## 第 2 章 Visual Basic 程序设计基础

本章介绍 VB 应用程序的基本元素，包括数据类型、常量、变量、表达式、函数等。

### 2.1 基本数据类型

描述客观事物的数、字符及所有能输入到计算机中并被计算机程序加工处理的符号的集合称为数据。数据是计算机程序处理的对象，也是运算产生的结果。

为了更好地处理各种各样的数据，VB 定义了多种数据类型。基本（标准）数据类型是系统预定义的数据类型，表 2-1 中列出了 VB 中定义的全部基本数据类型。

表 2-1 VB 的全部基本数据类型

数据类型	名称	符号	推荐前缀	占用字节数	取值范围
整型	Integer	%	int	2	-32 768~32 767
长整型	Long	&	lng	4	-2 147 483 648~2 147 483 647
单精度	Single	!	sng	4	负数：-3.402 823E38—1.401 298E-45 正数：1.401 298E-45~3.402 823E38
双精度	Double	#	dbl	8	负数：-1.797 693 134 862 32D308—4.490 656 458 41247D-324 正数：4.490 656 458 412 47D-324~1.797 693 134 862 32D308
日期型	Date(time)		dtm	8	
字符型	String	\$	str	取决于字符串长度	0~65 535 个字符
逻辑型	Boolean		bln	2	True 或 False（真或假）
货币型	Currency	@	cur	8	- 922 337 203 685 477.580 8~922 337 203 685 477.580 7
变体型	Variant		vnt	根据需要分配	
对象型	Object		obj	4	任何对象引用
字节型	Byte		byt	1	0~255

不同类型的数据，所占的存储空间不一样，使用合适的数据类型，可以优化代码。另外，数据类型不同，对其处理的方法也不同，这就需要数据类型的说明或定义。只有相同（相容）类型的数据之间才能进行操作，否则会出现错误。

VB 除提供了系统定义的基本数据类型外，还允许程序员根据需要定义自己的数据类型。下面介绍常用的基本数据类型。

#### 1. 数值（Numeric）型数据

在 VB 中，数值型数据分为整型数和浮点数两类。其中整型数又分为整数和长整数，浮点数分为单精度浮点数和双精度浮点数。

##### （1）整型数

整型数是不带小数点和指数符号的数，可以是正整数、负整数或 0。

① 整数 (Integer)

常规整型数简称为整数，用于表示不带小数点和指数符号的数，其占用字节数和取值范围见表 2-1。

整型数的运算速度较快，而且比其他数据类型占据的内存要少。在 For...Next 循环内作为计数器变量使用时，整数尤为有用。

十进制整数只能包含数字 0~9 和正负号（正号可以省略），十进制整数的取值范围为 -32 768~+32 767。例如：

168            12345            -286            0            6

十六进制整数由数字 0~9、字母 A~F（或 a~f）组成，并以 &H 引导，其后面的数据位数小于等于 4 位，其取值范围为 &H0~&HFFFF。

八进制整数由数字 0~7 组成，并以 &O 或 % 引导，其后面的数据位数小于等于 6 位，其取值范围为 &O0~&O177777。

② 长整型数 (Long)

长整型数的数字组成与整数相同，正号可以省略，并且在数值中不能出现逗号（分节符）。长整型数占用字节数和取值范围见表 2-1。

十进制长整数的范围为 -2 147 483 648~+2 147 483 647。例如：

32768            -12345678            -69            2147483647

十六进制长整型数以 &H 开头，以 & 结尾，其范围为 &H0~&HFFFFFFFF&。

八进制长整型数以 &O 或 % 开头，以 & 结尾，其范围为 &O0~&O37777777777&。

(2) 浮点数

浮点数也称实型数或实数，是带有小数部分的数。根据表示数值范围大小的不同，浮点数分为单精度数和双精度数。

① 单精度数 (Single)

单精度数用来表示带有小数部分的实数，可以精确到 7 位十进制数，小数点可以位于这些数字的任何位置，正号可以省略。单精度数有两种表示方法：定点表示法和浮点表示法。

- 单精度数的定点表示法

在单精度数表示的范围内，如果这个数值含有小数，而不含指数，则可用定点表示法。

例如：

-2.3            123.4            +1.234            0.0000568            -687.654321

- 单精度数的浮点表示法

浮点表示法就是数学中的科学计数法，即以 10 的整数次幂表示的数，以字母 “E” 来表示底数 10。例如， $-1.6 \times 10^9$ ， $123.4 \times 10^{-12}$ ， $+34.56 \times 10^{28}$ ， $0.000\,987\,65 \times 10^{-20}$  分别表示为：

-1.6E9            123.4E-12            34.56E+28            0.00098765E-20

② 双精度数 (Double)

双精度数可以精确到 15 位或 16 位十进制数，小数点可以位于这些数字的任何位置，正号可以省略。双精度数也有两种表示方法：定点表示法和浮点表示法。

- 双精度数的定点表示法

在双精度数表示的范围内，如果这个数含有小数，而不含指数，则可用定点表示法。

例如：



-12.123456789123      0.987654321      100000000.1234

- 双精度数的浮点表示法

这是更大范围的科学计数法，即以 10 的整数次幂表示的数，以字母“D”来表示底数 10。例如：

-1.234567D92      123.123456789D-45      0.123456789D+5

### (3) 数值型数据的使用说明

在 VB 中，声明和使用数值型数据时，应注意以下 4 点。

① 如果数据包含小数，则应使用单精度数或双精度数。

② 在 VB 中，数值型数据都有一个有效的范围值，程序中的数如果超出规定的范围，就会出现“溢出”信息（Overflow）。如果该数小于范围的下限值，系统将按 0 处理；如果大于上限值，则系统只按上限值处理，并显示出错误信息。

③ 一般，VB 使用十进制数计数，但有时也使用十六进制数和八进制数表示，表示值时它们与十进制数是等价的。

④ 所有数值变量都可相互赋值，也可对变体（Variant）类型变量赋值。在将浮点数赋予整数之前，VB 要将浮点数的小数部分四舍五入，而不是将小数部分去掉。

## 2. 字符（String）型数据

字符型数据是一个字符序列，字符型数据常简称为字符串。字符串是放在两个双引号(")之间的若干个字符。在 VB 中，一个英文字母或一个汉字都被认为是一个字符，都占用两个字节。长度为 0（即不含任何字符）的字符串称为空字符串。字符串允许的最大长度见表 2-1。

在 VB 中，字符串有两种：变长字符串和定长字符串。

### (1) 变长字符串

变长字符串是指长度不固定的字符串。如果对字符串变量赋予新的字符串，它的长度也随之增减。按照默认规定，一个字符串如果没有定义成固定长度，都属于可变长字符串。例如：

"Visual Basic 6.0"      "6+8"      "中国"      "20071026"

### (2) 定长字符串

定长字符串是指在程序执行过程中，始终保持其长度不变的字符串。例如，定义一个长度为 8 个字符的字符串变量：当赋予字符串的字符少于 8 个时，用空格将不足部分填满；当赋予字符串的字符长度超过 8 个时，截去超出部分的字符。

## 3. 逻辑（Boolean）型数据

逻辑型（或称布尔型）数据只有两个值：真（True）和假（False），常被用来表示逻辑判断的结果。任何只有两种状态的数据，如 True/False, Yes/No, On/Off 等，都可以表示为逻辑型。

当把数值型数据转换为逻辑型时，0 会转换为 False，其他非 0 值转换为 True。当把逻辑型转换为数值型时，False 转换为 0，True 转换成-1。

## 4. 日期（Date）型数据

日期型数据用来表示日期和时间，可以表示多种格式的日期和时间，表示的日期范围从

公元 100 年 1 月 1 日到 9999 年 12 月 31 日，而时间可以从 0:00:00 到 23:59:59。日期型数据用两个“#”符号把表示日期和时间的值括起来，就像字符串数据是用双引号括起来的一样。其格式为 mm/dd/yyyy 或 mm-dd-yyyy，也可以为 yyyy/mm/dd。如果带有时间，其格式为 mm/dd/yyyy hh:mm:ss AM|PM 或 mm-dd-yyyy hh:mm:ss AM|PM。例如：

```
#03/30/2007#           #2007-03-30#           #03/30/2007 07:25:12 AM#
```

如果输入的日期或时间非法或不存在，系统将显示出错信息。

将其他数据类型的数值转换为日期型时，小数点左边的值代表日期信息，小数点右边的值则代表时间信息。0 为午夜，0.5 为正午。负数表示公元 1899 年 12 月 31 日之前的日期。

日期的具体显示格式，由 Windows 控制面板中的“区域设置”项决定。

5. 变体 (Variant) 型数据

Variant 型的数据能够表示所有系统定义类型的数据，当把它们赋予 Variant 型数据时，不必进行类型转换，VB 会自动完成任何必要的转换。

在程序中不特别说明时，VB 会自动将该变量默认为 Variant 型变量，例如：

```
a = "6"                ' a 的值为字符型数据 "6"
a = 6 - 2               ' a 的值为数值型数据 4
a = "D" & a             ' a 的值为字符型数据 "D4"
SomeValue = "18"        ' SomeValue 包含 "18" (字符串)，字符型
SomeValue = SomeValue - 15 ' 现在 SomeValue 值为 3，数值型
SomeValue = "U" & SomeValue ' 现在 SomeValue 包含 "U3" (字符串)，字符型
```

要尽量少用 Variant 数据类型，以避免发生错误。如果对 Variant 型变量进行数学运算或函数运算，则该变量必须包含某个数。要连接两个字符串，则应该用“&”操作符，而不要用“+”操作符。

6. 其他数据类型

在 VB 中，除了前面介绍的数据类型外，还有其他一些数据类型。

(1) 字节 (Byte) 型数据

Byte 型数为无符号的整数，占用 1 字节 (Byte)，范围为 0~255。除一元减法外，所有可对整数进行操作的运算符均可对 Byte 型数据操作。因为 Byte 型数是 0~255 的无符号类型数，所以不能表示负数。因此，在进行一元减法运算时，VB 首先将 Byte 转换为整数。

(2) 货币 (Currency) 型数据

Currency 型数据是为表示钱款而设置的，Currency 型数占用的字节数和取值范围见表 2-1。Currency 型支持小数点右边 4 位和小数点左边 15 位，它是一个精确的定点数据类型，适用于货币计算。浮点数比 Currency 型数的有效范围大得多，但有可能产生小的进位误差。

浮点数中的小数点是“浮动”的，即小数点可以出现在数的任何位置，而货币型数据的小数点是固定的，因此称为定点数据类型。

(3) 对象 (Object) 型数据

对象型数据用来表示图形、OLE 或其他对象。

# 2.2 变量

在程序中处理数据时，对于输入的数据、参加运算的数据、运行结果等临时数据，通常将它们暂时存储在计算机的内存中。变量就是命名的内存单元位置。一旦定义了某个变量，该变量表示的都将是同一个内存位置。程序员使用变量名，就可在程序的其他部分引用该内存位置，直到释放该变量。

对于变量，在程序执行的每个瞬间，变量的值都是确定的、已知的；但在程序执行的过程中，它的值是可以变化的。可以形象地理解为：一个变量就是一个盒子，盒子有一个名字，盒子中存放的东西就是数据。一个变量在一个时刻只能存放一个值。变量中存放的数据称为“变量的当前值”（简称变量值），且每个变量都有数据类型。如果某一个变量在程序运行中数据发生变化，则当前值将取代原来的值。例如，将整型数 6 放到变量 a 中，则原来的数被清除，当前值 6 就成为变量 a 的值。

变量有两个特性：名字和数据类型。变量的名字用于在程序中标识变量和使用变量的值，数据类型则确定了变量中可以保存哪种数据。

在 VB 中，变量有两种形式：属性变量和内存变量。

在窗体中设计程序界面时，VB 会自动为产生的对象（包括窗体本身）创建一组变量，即属性变量，并为每个变量设置其默认值。这类变量可供程序员直接使用，如引用其值或赋予新值。由于属性变量是 VB 系统自动创建的，所以无须程序员费心。内存变量则要靠程序员根据程序需要创建，下面主要介绍内存变量的建立方法。

## 1. 变量的命名规则

变量代表在程序执行过程中其值可以改变的存储单元，这个存储单元的名字称为变量名。VB 变量名的命名规则如下。

（1）变量名由 1~255 个字符组成，包含的字符可以是数字、英文字母或下划线，中间不能有“.”或其他类型说明字符，并且必须以英文字母开头。例如：

xm	ab2	i	name	是合法的变量名
int.sum	（因其中有小数点）	2ab		是非法的变量名

（2）不能用 VB 的关键字作为变量名，但可以把关键字嵌入变量名中；同时，变量名也不能是末尾带有类型说明符的关键字。例如，变量名 Print 和 Print\$ 是非法的，而变量名 Print\_Number 是合法的。

（3）变量名在同一个范围内必须是唯一的。

## 2. 变量命名的注意事项

（1）最好使用有明确实际意义和容易记忆的、通用的变量名，要见名知义。例如，用 Sum（或 s）代表求和，用 Difference（或 d）代表求差等。

（2）尽可能简单明了，尽量不要使变量名太长，因为太长不便阅读和拼写。

（3）不能用 VB 的关键字作为变量名。VB 的关键字是指 VB 中系统已经定义的词，如语句、函数、运算符名等。

（4）变量名不能与过程名和符号常量名相同。

(5) 尽量采用 VB 建议的变量名前缀或后缀的约定来命名, 以便区分变量的类型, 如: intMax, strName。

(6) VB 不区分变量名和其他名字中字母的大小写, 如 Hello, HELLO, hello 指的是同一个名字。

3. 变量声明

使用变量前, 一般应先声明变量名及其类型, 以使系统为其分配存储单元。

与其他语言不同, VB 不要求程序员在使用变量前特别声明。如果没有声明变量, VB 就使用称为变体 (Variant) 类型的默认数据类型。然而, 使用可变类型存储通用信息有两个缺点: 第一, 它会浪费内存空间; 第二, 在与某些数据处理功能同时使用时, 可变类型可能无效。所以, 在使用变量前最好先声明变量, 把将要用到的数据类型告诉程序。

(1) 声明变量

所谓声明变量, 就是用一个语句来定义变量的类型, 又称为显式声明。声明变量语句并不把值分配给变量, 而是告知变量将会包含的数据。声明语句的语法为:

```
{Dim | Private | Static | Public} <变量名> [As <类型>] [, <变量名 2> [As <类型 2>] || ...
```

【说明】

① Public 语句用来声明公有的模块级变量, Private 语句或 Dim 语句用来声明私有的模块级变量, Dim 语句、Private 语句或 Static 语句用来声明过程级局部变量 (参见第 9 章)。

② <变量名> 遵循标准的变量命名约定。

③ <类型> 用来定义被声明 <变量名> 的数据类型或对象类型。变量的数据类型可以是表 2-1 中的类型, 也可以是程序员自定义的类型。例如:

```
Dim Count As Integer
Private Sum As Single, strName As String
Static Average As Single
Public Yn As Boolean
Private Name1 As String*8
Dim aa ' 若没有指定类型, 变量是 Variant 型
```

使用声明语句建立一个变量后, VB 自动将数值型的变量赋初值 0, 将字符型或 Variant 型的变量赋空串, 将布尔型的变量赋 False。

使用变量时, VB 会自动转换变量值的类型, 使变量的值与声明语句中的名字相匹配。例如, 声明变量为:

```
Dim Count As Integer
当为该变量赋值时:
Count = 1.5 ' 数 1.5 为单精度浮点数
```

变量 Count 自动将 1.5 转换为整型数 (Integer) 2 (四舍五入)。

(2) 强制显式声明变量语句 Option Explicit

声明变量可以有效地降低错误率。为了避免写错变量名引起的麻烦, 可以规定在使用变量前, 必须先用声明语句进行声明, 否则 VB 将发出警告 “Variable not defined” (变量未定义)。要强制显式声明变量, 可以在类模块、窗体模块或标准模块的声明段中加入语句:

### Option Explicit

也可以执行“工具”框→“选项”命令，在打开的“选项”对话框中单击“编辑器”选项卡，再选中“要求变量声明”复选框，如图 2-1 所示。之后，VB 会在后续的窗体模块、标准模块及类模块中自动插入 Option Explicit，这一语句总是显示在代码编辑窗口的顶部，如图 2-2 所示。

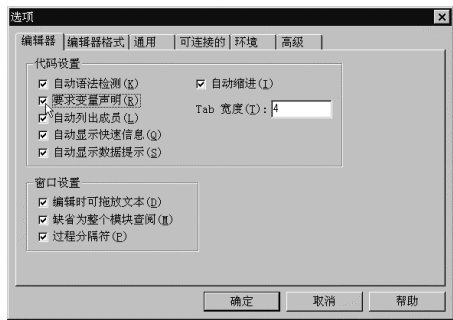


图 2-1 “选项”对话框中的“编辑器”选项卡

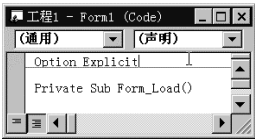


图 2-2 代码窗口

但是，Option Explicit 语句的作用范围仅限于语句所在模块，所以，对每个需要强制显式声明变量的窗体模块、标准模块及类模块，必须将 Option Explicit 语句放在这些模块的声明段中。

注意，VB 不会将 Option Explicit 语句加入到已有代码中，必须在工程中手工将该语句加入。方法是：激活代码编辑窗口，从对象下拉列表框中选择“通用”项，从过程下拉列表框中选择“声明”项，在图 2-2 所示的位置输入“Option Explicit”。

## 2.3 常量

常量是指在程序运行过程中始终保持不变的常数、字符串等。在 VB 中，有两种形式的常量：直接常量和符号常量。

### 1. 直接常量

直接常量是在程序代码中，以直接明显的形式给出的数据。根据使用的数据类型，可分为：字符串常量、数值常量、逻辑常量和日期常量。

#### (1) 字符串常量

字符串常量是用两个双引号括起来的一串字符。这些字符可以是除双引号和回车符、换行符以外的所有字符。例如：

"\$3,123.123"          "3859987"          "Visual"

#### (2) 数值常量

数值常量就是常数，包括整数、长整数、单精度数、双精度数。例如：

123          987654321          3.1415926          5.8D80

#### (3) 逻辑常量

逻辑常量只有 True（真）和 False（假）两个值。

#### (4) 日期常量

用两个“#”号把表示日期和时间的值括起来表示日期常量。例如：

#03/30/2007#          #30/03/2007#          #May 30 2007 21:47:12#

## 2. 符号常量

如果在程序中多次出现一些很大的数字或很长的字符串，为了改进代码的可读性和可维护性，可以给某一特定的值赋予一个名字，以后用到这个值时就用此名字代表，这样便于程序修改和阅读，这个名字就是符号常量。

符号常量有点像变量，但不能像对变量那样修改符号常量，也不能对符号常量赋以新值。符号常量分为两种：系统提供的常量和程序员声明的常量。

#### (1) 系统提供的常量

系统提供的常量是指 VB 内置的一些便于记忆的常量。为了避免不同对象库中同名常量的混淆，在引用时可使用两个小写字母作为前缀来限定在哪个对象库中，例如：

vb: 表示 VB 和 VBA 中的常量，如 vbModal 代表 1，vbModalless 代表 0

db: 表示 Data Access Object 库中的常量

xl: 表示 Excel 中的常量

通过使用系统常量可以使程序更加易于阅读和理解。例如，窗体的状态属性 WindowsState 可以接受系统常量 vbNormal(0)、vbMinimized(1)和 vbMaximized(2)。显然，语句

```
Form1.WindowsState=vbMaximized
```

要比

```
Form1.WindowsState=2
```

更易理解。

#### (2) 程序员声明的常量

尽管 VB 内部定义了大量的常量，但是有时程序员还需要创建自己的符号常量。可以使用 Const 语句来定义常量，并给常量分配名字、值和类型。声明常量的语法格式为：

**[Public | Private] Const <常量名> [As <数据类型>] = <表达式> ...**

#### 【说明】

① <常量名> 的命名规则与变量名的规则一样。

② <表达式> 由数值、字符串等常量及运算符组成，可以包含前面定义过的常量，但不能使用函数调用，例如：

```
Const MAX As Integer = 100                    ' 声明常量 MAX，代表 100，整型数
```

```
Const PI = 3.14159                           ' 声明常量 PI，代表 3.14159，单精度数
```

```
Const XH = "20071101"                       ' 声明常量 XH，代表"20071101"，字符串
```

③ 如果用逗号进行分隔，则在一行中可放置多个常量声明：

```
Const PI = 3.14159 , MAX = 100 , XH = "20071101"
```

## 2.4 运算符和表达式

运算是 对数据进行加工的过程。描述各种不同运算的符号称为运算符，而参与运算的数据称为操作数。表达式用来表示某个求值规则，它由运算符和配对的圆括号将常量、变量、

函数、对象等操作数以合理的形式组合而成。

表达式可用来执行运算、操作字符或测试数据，每个表达式都产生唯一的值。表达式的类型由运算符的类型决定。在 VB 中有 5 类运算符和表达式：算术运算符和算术表达式、字符串运算符和字符串表达式、日期运算符和日期表达式、关系运算符和关系表达式、逻辑运算符和逻辑表达式。下面分别介绍。

### 2.4.1 算术运算符和算术表达式

#### 1. 算术运算符

算术运算符是最常用的运算符，用来进行简单的算术运算。VB 有 7 个算术运算符，见表 2-2。

表 2-2 算术运算符及其含义（按优先级排列）

运 算 符	含 义	优 先 级	示 例
^	指数运算	1	2 ^ 3，值为 8
*	乘	2	2 * 3，值为 6
/	浮点除	3	1 / 2，值为 0.5
\	整数除	4	1 \ 2，值为 0
Mod	求余的模运算	5	1 Mod 2，值为 1
+	加	6	1 + 2，值为 3
-	减、取负	7	3 - 2，值为 1 -1，值为-1

【说明】

- ① 在这 7 个算术运算符中，只有取负“-”是单目运算符，其他均为双目运算符。
- ② 加“+”、减“-”、乘“\*”、浮点除“/”、取负“-”、指数运算“^”的含义与数学中的基本相同。
- ③ “/”和“\”的区别：1 / 2 = 0.5，1 \ 2 = 0。整除号“\”用于整数除法（简称整除），在进行整除时，如果参加运算的数据含有小数，首先将它们四舍五入，使其成为整型数或长整型数，然后再进行运算，其结果截尾成整型数。
- ④ 模运算符 Mod 用来求整除的余数，其结果为第一个操作数整除第二个操作数后所得的余数。例如，7.8 Mod 3.4，首先把 7.8 和 3.4 分别取整为 8 和 3，再进行整除，余数为 2。
- ⑤ 进行除法（包括整除）运算时除数为 0，或者进行乘幂运算时指数为负数而底数为 0，都会产生溢出的错误信息。

#### 2. 算术表达式

算术表达式也称数值型表达式，由算术运算符、数值型常量、变量、函数和圆括号组成，其运算结果为一个数值。例如，2 \* 4 + 10 的运算结果为 18.00。算术表达式的格式为：

〈数值 1〉〈算术运算符 1〉〈数值 2〉[ 〈算术运算符 2〉〈数值 3〉] …

#### 3. 算术表达式的书写规则

算术表达式与数学中的表达式写法有所区别，在书写时应当特别注意以下事项。

- ① 每个符号占 1 格，所有符号都必须一个一个地并排写在同一条横线上，不能在右上角或右下角写方次或下标。例如， $2^3$  要写成  $2^3$ ， $x_1+x_2$  要写成  $x1+x2$ 。
- ② 在数学表达式中省略的内容必须重新写上。例如， $2xy$  要写成  $2*x*y$ 。
- ③ 所有括号都用圆括号，括号必须配对。例如， $2[x+6(y+z)]$  必须写成  $2*(x+6*(y+z))$ 。
- ④ 要把数学表达式中的有些符号改成 VB 中可以表示的符号。例如，要把  $\pi r^2$  改为  $PI*r^2$ 。

4. 算术运算符的优先级

算术表达式中包含各种算术运算符时，必须规定各个运算的先后顺序，这就是算术运算符的优先级，如下所示：

指数运算(^)→取负(-)→乘(\*)、浮点除(/)→整除(\)→求模 Mod →加(+)、减(-)

其中，乘和浮点除是同级运算符，加和减是同级运算符。当一个表达式中含有多种算术运算符时，将按上述顺序求值。如果表达式中含有圆括号，则先计算括号中表达式的值；如果表达式中有多层括号，先计算最内层括号中表达式的值。

5. 算术表达式中数据类型的匹配规则

在算术表达式中，允许不同类型的数值数据参与运算，VB 对其运算结果的类型做了以下规定。

- ① 同类型数据运算后，其结果的类型保持不变。
- ② 整型数与实型数运算后，其结果为实型数。
- ③ 整型数与长整型数运算后，其结果为长整型数。
- ④ 单精度数与双精度数运算后，其结果为双精度型数。

总之，运算结果的类型服从其中“级别”较高的类型。其级别高低（从低到高）顺序为：

整型（2 字节）→长整型和单精度型（4 字节）→双精度型（8 字节）

2.4.2 字符串运算符和字符串表达式

1. 字符串运算符

VB 中的字符串运算符是“&”，该运算符用于连接两个或更多的字符串。

2. 字符串表达式

字符串表达式由字符串常量、字符串变量、字符串函数和字符串运算符组成。它可以是一个简单的字符串常量，也可以是若干个字符串常量或字符串变量的组合。字符串表达式的格式为：

〈字符串 1〉 & 〈字符串 2〉 [& 〈字符串 3〉] …

当两个字符串用连接运算符“&”连接起来后，第二个字符串直接添加到第一个字符串的尾部，结果是一个更长的、包含两个源字符串的全部内容的字符串。要把多个字符串连接起来，每两个字符串之间都要用“&”来分隔。例如：

"1234" & "56" ' 连接后结果为: "123456"



"Visual " & "Basic" & " 6.0" ' 连接后结果为: "Visual Basic 6.0"

另外，在 VB 中，除用“&”作为连接运算符外，还可以用“+”把两个字符串连接成一个字符串。但是“+”容易与算术加法运算符产生混淆，所以，建议最好用“&”。同时，“&”会自动将非字符串类型的数据转换成字符串后再连接，“+”则不能自动转换。例如：

123 & 456 & "abcd" ' 连接后结果为: "123456abcd"

2.4.3 日期运算符和日期表达式

1. 日期运算符

日期型数据是一种特殊的数值型数据，它们之间只能进行加“+”、减“-”运算。

2. 日期型表达式

日期型表达式由算术运算符（“+”或“-”）、算术表达式、日期型常量、日期型变量和函数组成。日期型表达式的运算有下面三种情况。

① 两个日期型数据相减，结果是一个数值型数据（两个日期相差的天数），例如：

#09/16/2007#-#09/11/2007# ' 结果为数值型数据: 5

② 一个表示天数的数值型数据加到日期型数据中，其结果仍然为一个日期型数据（向后推算日期），例如：

#09/16/2007#+12 ' 结果为日期型数据: #07/09/28#

③ 一个表示天数的数值型数据从日期型数据中减掉它，其结果仍然为一个日期型数据（向前推算日期），例如：

#09/16/2007#-10 ' 结果为日期型数据: #07/09/06#

2.4.4 关系运算符和关系表达式

1. 关系运算符

关系运算符（又称关系比较符）用来对两个表达式的值进行比较，比较的结果是一个逻辑值，即真（True）或假（False）。VB 提供了 8 个关系运算符，见表 2-3。

表 2-3 关系运算符及其含义

关系运算符	含 义	例 子	运算结果
=	等于	1+4=5	True
>	大于	5*4>20	False
<	小于	"A"<"B"	True
>=	大于或等于	1.5+2>=4.5	False
<=	小于或等于	7-5<=8*6	True
<>	不等于	4*3<>5+5	True
Like	比较样式		
Is	比较对象变量		

在使用的时候要注意，关系运算符的写法与数学中的写法不同。

## 2. 关系表达式

关系表达式由关系运算符、算术表达式、字符串表达式、日期表达式，或者作为表达式特例的常量、变量、函数组成，但关系运算符两侧的数据类型必须完全一致。关系表达式的格式为：

〈表达式 1〉〈关系运算符〉〈表达式 2〉

关系运算符用于对两个相同数据类型的表达式值的大小进行比较，被比较的数据可以是数值型数据、字符型数据、日期型数据，但不能是逻辑型数据，其运算结果为逻辑型数据：真（True）或假（False）。

在书写含有两个字符的关系运算符时，它们之间不能有空格。

表 2-3 中前 6 个运算符可用于两个数值型，或者两个字符型，或者两个日期型数据的比较。而后两个运算符（Like, Is）用于专门的字符型数据的比较。

关系运算符都是单独使用的，不存在优先级问题。关系表达式的运算次序为：先分别运算关系运算符两侧的表达式，然后把二者进行比较。二者的关系若与运算符指示的一样，则运算结果为 True，否则为 False。关系表达式通常作为语句中的一个条件。

当数值作为条件出现在语句中时，用 0 代表假，任何非 0 数都代表真（但一般用 -1 表示真）。

## 3. 使用关系比较时的注意事项

（1）数值型数据按其数值大小进行比较。对于单精度数或双精度数，在进行“=”（等于）比较时要特别小心，运算可能会产生误差而得出非常接近但不相等的结果。例如：

$1.0/6.0*6.0=1.0$

上式在数学上是一个恒等式，但在计算机上执行时有可能会给出 False。因此，应避免对两个浮点数做“相等”或“不相等”的判断。上式可改写为：

$Abs(1.0/6.0*6.0-1.0)<1E-5$  （Abs 是求绝对值的函数）

只要它们的差小于一个很小的数（这里是  $10^{-5}$ ），就认为它们相等。

（2）对于字符型数据，若是单个字符，则按其 ASCII 码值大小进行比较；若是汉字字符，则按内码顺序进行比较。比较字符串时，从关系符左边字符串的第一个字符开始，逐一与右边字符串中对应位的字符进行比较，最先发现不一样的字符彼此是什么关系，相应字符串之间就是什么关系。若两个字符串的长度不等，则短字符串可以采用尾部补空格的方法补齐。常见字符值的大小顺序排列如下：

" "<"0"<...<"9"<"A"<...<"Z"<"a"<...<"z"<"任何汉字"

（3）日期型数据将日期看成“yymmdd”格式的 6 位整数，按数值大小进行比较。

（4）Like 运算符用来比较字符串表达式和 SQL 表达式中的样式，主要用于数据库查询。Is 运算符用来比较两个对象的引用变量，主要用于对象操作。此外，Is 运算符还在 Select Case 语句中使用。

### 2.4.5 逻辑运算符和逻辑表达式

一个简单的条件可以用一个关系表达式来表示，如果要指定的是一个由几个简单条件组成的复合条件，例如， $x>8$  且  $x<20$ ，在数学上可以表示为： $8<x<20$ ，而在 VB 语言中就必须先写成两个关系表达式，再由一个逻辑运算符把它们连接起来，即  $x>8$  And  $x<20$ 。

1. 逻辑运算符

逻辑运算（也称布尔运算）用逻辑运算符连接两个或多个关系式，组成一个逻辑表达式。VB 中的逻辑运算符有 6 个，见表 2-4。

表 2-4 逻辑运算符及其含义

逻辑运算符	含 义	优 先 级	例 子	运算结果
Not	逻辑非	1	Not 3>2	False
And	逻辑与	2	3<=6 And 5>2	True
Or	逻辑或	3	3<=6 Or 2>5	True
Xor	异或	4	3>8 Xor 5<6	True
Eqv	等价	5	3>=6 Eqv 2>5	True
Imp	蕴涵	6	3>=6 Imp 2>5	True

2. 逻辑表达式

逻辑表达式由关系表达式、逻辑运算符、常量、变量和函数组成。逻辑表达式的一般格式为：

〈关系表达式 1〉〈逻辑运算符〉〈关系表达式 2〉

逻辑运算符用来对逻辑型数据进行各种逻辑运算。逻辑表达式运算的结果与关系表达式相同，仍然为逻辑型数据，即 True 或 False。各运算符使用方法介绍如下。

- Not: 单目运算，由真变假或由假变真，进行取反运算。例如，Not 2>5 的值为 True。
- And: 对两个关系式进行比较，如果两个关系表达式的值均为 True，则结果为 True；否则为 False。例如，2<5 And 8>3 的值为 True。

Or: 对两个关系表达式进行比较，如果其中有一个表达式的值为 True，结果就为 True；只有两个表达式的值均为 False 时，结果才为 False。例如，2>5 Or 8>3 的值为 True。

Xor: 对两个关系式进行比较，如果两个表达式同时为 True 或同时为 False，则结果为 False；否则为 True。例如，2>5 Xor 8>3 的值为 True，2>5 Xor 8<3 的值为 False。

Eqv: 对两个关系式进行比较，如果两个表达式同时为 True 或同时为 False，则结果为 True；否则为 False。例如，2>5 Eqv 8>3 的值为 False，2>5 Eqv 8<3 的值为 True。

Imp: 对两个关系式进行比较，当第一个表达式为 True，且第二个表达式为 False 时，结果为 False；否则为 True。例如，5>2 Imp 8<3 的值为 False，2>5 Imp 8<3 的值为 True。

表 2-5 中列出了每一种逻辑运算可能返回的结果。

表 2-5 逻辑表达式真值表

表达式 1	表达式 2	Not	And	Or	Xor	Eqv	Imp
True	True	False	True	True	False	True	True
True	False	False	False	True	True	False	False
False	True	True	False	True	True	False	True
False	False	True	False	False	False	True	True

3. 逻辑表达式的运算顺序

一个逻辑表达式内可能包括逻辑运算符、关系运算符或算术运算符，在 VB 中规定其优

先级别为：算术运算→关系运算→逻辑运算。例如：

2+3>5 And 5<3

结果为 False

Not 5<3

结果为 True

5>=5 Or 4\*7<>7

结果为 True

注意，关系表达式绝不能比较逻辑型数据！例如，若 Yn 为布尔型变量，则下面写法是错误的：

Yn = True

（这里的“=”是关系运算符“等于”，不是赋值语句）

2.4.6 表达式的执行顺序

一个表达式中可能含有多种运算，例如：

Abs(-5)\*2+8>=9\*8 Or "ABC"="AB" And Not 3-2<3\*5

VB 将按一定的顺序对表达式求值。一般运算顺序如下：

- 第一级 函数运算
- 第二级 算术运算    ^ → -（取负） → \*, / → \ → Mod → +, -
- 第三级 关系运算    <, <=, =, >=, >, <>
- 最后级 逻辑运算    Not → And → Or → Xor → Eqv → Imp

所有同一级运算都是从左到右进行的，括号内的运算优先执行，嵌在最内层括号里的运算优先级最高，然后依次由内向外执行。

运算符的优先顺序见表 2-6。

表 2-6 运算符的优先顺序

优 先 顺 序	运算符类型	运 算 符
1	算术运算符	^（指数）
2		-（取负）
3		*,/（乘和除）
4		\（整除）
5		Mod（求模）
6		+, -（加和减）
7	字符串运算符	&（字符串连接）
8	关系运算符	=, <>, <, >, <=, >=
9	布尔运算符	Not
10		And
11		Or

【说明】

- ① 当乘与除运算符同时出现在表达式中时，将按照从左到右出现的顺序依次计算。用括号可以改变表达式的优先顺序。
- ② 字符串连接运算符(&)不是算术运算符，就其优先顺序来说，它在所有算术运算符之后，而在所有关系运算符之前。
- ③ 上述运算顺序有一个例外，就是当幂与负号相邻时，负号优先。例如，2^-3 的运算

结果是 0.125 ( $2^{-3} = \frac{1}{2^3}$ )。

**【例 2-1】** 求 VB 表达式  $4 + 2 > 3 + 5 \text{ And Not } 2 < 3$  的值。  
**【分析】** 在计算前，先要看清表达式中有哪些运算符，再根据运算符的优先级进行计算。本例应按下面的步骤进行计算。

- ① 算术运算                       $6 > 8 \text{ And Not } 2 < 3$
- ② 关系运算                       $\text{False And Not True}$
- ③ 非运算                         $\text{False And False}$
- ④ 结果                           $\text{False}$

**【例 2-2】** 一元二次方程  $ax^2 + bx + c = 0$  有实根的条件为： $a \neq 0$ ，并且  $b^2 - 4ac \geq 0$ ，写出相应的 VB 逻辑表达式。

**【分析】** 一元二次方程  $ax^2 + bx + c = 0$  有实根的条件有两个，即  $a \neq 0$  和  $b^2 - 4ac \geq 0$ 。其中， $a \neq 0$  用 VB 表达式表示为： $a <> 0$ ； $b^2 - 4ac \geq 0$  用 VB 表达式表示为： $b^2 - 4 * a * c >= 0$ 。二者是与（And）的关系，用 And 连接上面的两个式子，结果为：

$$a <> 0 \text{ And } b^2 - 4 * a * c >= 0$$

**【例 2-3】** 闰年的条件是：年份能被 4 整除，但不能被 100 整除；或者能被 400 整除。写出闰年的 VB 逻辑表达式。

**【分析】** 设 y 表示年份。被某个数整除，可以用求模运算符 Mod 或 \ 或 Int() 函数来实现。年份能被 4 整除，但不能被 100 整除的表达式为：

$$y \text{ Mod } 4 = 0 \text{ And } y \text{ Mod } 100 <> 0$$

年份能被 400 整除的表达式为：

$$y \text{ Mod } 400 = 0$$

两者取“或”，即得判断闰年的逻辑表达式：

$$(y \text{ Mod } 4 = 0 \text{ And } y \text{ Mod } 100 <> 0) \text{ Or } (y \text{ Mod } 400 = 0)$$

用 Int() 函数表示为：

$$(\text{Int}(y/4) = y/4 \text{ And } \text{Int}(y/100) <> y/100) \text{ Or } (\text{Int}(y/400) = y/400)$$

## 2.5 常用内部函数

在 VB 中有两类函数，即内部函数和程序员定义函数。内部函数也称标准函数，VB 提供了大量的内部函数。程序员定义函数是由程序员根据需要定义的函数。

函数是一种特定的运算。在程序中要使用一个函数时，只需给出函数名并给出一个或多个参数，就能得到函数值。VB 提供了大量的内部函数供程序员在编程时调用，这些内部函数实际上是事先编好的一些程序模块，被封装在 VB 内部。程序员在使用时可以不考虑其内部结构，而只关心它的输入与输出状况。这些函数按功能可以分为：数学函数、转换函数、字符串函数、日期函数和格式函数等。下面介绍一些常用的内部函数。

### 2.5.1 数学运算函数

数学运算函数用于各种数学运算，与数学中的定义基本相同，其中常用数学函数的名称

及简要说明见表 2-7。

表 2-7 常用数学函数

函 数 名	说 明	示 例	结 果
Sin(N)	返回弧度的正弦	Sin(0)	0
Cos(N)	返回弧度的余弦	Cos(0)	1
Tan(N)	返回弧度的正切	Tan(0)	0
Atn(N)	返回用弧度表示的反正切值	Atn(0)	0
Abs(N)	返回数的绝对值	Abs(-3.5)	3.5
Exp(N)	返回 e 的指定次幂，即 $e^N$	Exp(3)	20.086
Log(N)	返回一个数值的自然对数	Log(10)	2.3
Sqr(N)	返回数的平方根	Sqr(9)	3
Sgn(N)	返回数的符号值：正数返回 1，负数返回-1，0 返回 0	Sgn(-3.5)	-1
Int(N)	返回不大于给定数的最大整数	Int(-1.6)	-2
Fix(N)	返回数的整数部分	Fix(-1.6)	-1

【说明】

- ① 函数名中的 N 代表算术表达式。
- ② 在使用三角函数时，角度以弧度为单位。如果 N 为度，则要转换为弧度，例如，Sin(90\*(3.1415926/180))。
- ③ Int(N)取小于或等于 N 的最大整数。例如，Int(7.65)的函数值为 7，Int(-7.65)的函数值为-8。
- ④ Fix(N)的功能是舍去实数的小数部分。当  $N \geq 0$  时，Fix()和 Int()有相同的结果；而当  $N < 0$  时，结果不同。例如，Fix(7.65)的函数值为 7，Fix(-7.65)的函数值为-7。
- ⑤ 符号函数 Sgn(N)的定义为：当  $N > 0$  时，Sgn(N)的值为 1；当  $N = 0$  时，Sgn(N)的值为 0；当  $N < 0$  时，Sgn(N)的值为-1。例如，SGN(28)的函数值为 1，SGN(0)的函数值为 0，SGN(-28)的函数值为-1。

2.5.2 字符串函数

VB 提供了大量的字符串函数，具有强大的字符串处理能力，其中常用的字符串函数及简要说明见表 2-8。

表 2-8 常用字符串函数

函 数 名	说 明	示 例	结 果
InStr([N1,]C1,C2[M])	在 C1 中从 N1 开始查找 C2，省略 N1 则从头开始找，找不到为 0	InStr(2,"ABCDEFGH","EF")	5
InStrRev(C1,C2,[N1],[M])	与 InStr 函数作用相似，只是从字符串的尾部开始查找	InStr("ABCDEFGH","EF",2)	7
Join(A[,D])	将数组 A 中各元素用分隔符 D 连接成字符串变量	A=array("123","ab","c") Join(A,"")	"123abc"
Left(C,N)	取出字符串左边 N 个字符	Left("ABCDEFGH",3)	"ABC"
Len(C)	测量字符串的长度	Len("ABC 学习的开始")	8

续表

函 数 名	说 明	示 例	结 果
LenB(C)	测量字符串所占的字节数	LenB("AB 高等数学")	12
LTrim(C)	去掉字符串左边的空格	LTrim(" ABCD")	"ABCD"
Mid(C,N1,N2)	从 C 中 N1 位开始取出长度为 N2 的子串	Mid("ABCDEFG",3,2)	"CD"
Replace(C,C1,C2[,N1][,N2][,M])	在 C 字符串中从 N1 位开始用 C2 替换 C1 (有 N2 时替换 N2 次)	Replace("ABCDABCD","CD","123")	AB123AB123
Right(C,N)	取出字符串右边 N 个字符	Right("ABCDEFG",3)	"EFG"
RTrim(C)	去掉字符串右边的空格	RTrim("ABCD ")	"ABCD"
Space(N)	产生由 N 个空格组成的字符串	"ABC" & Space(2) & "123"	"ABC 123"
Split(C[,D][,N][,M])	将字符串 C 用分隔符 D 分隔成字符串组, 与 Join 函数的功能相反	S=Split("123,56,ab",",")	S(0)="123" S(1)="56" S(2)="ab"
StrComp(C1,C2[,M])	比较两个字符串的大小, 并以-1,0,1 分别表示两个字符串的大小	StrComp("ABCDEF","BC")	-1
String(N,C)	返回由 C 中 N 个首字符组成的字符串	String(3,"ABCDEF")	"AAA"
StrReverse(C)	将字符串反序	StrReverse("ABCDEF")	"FEDCBA"

注：表中的参数 M 表示是否区分大小写：省略 M 或 M=0 表示区分大小写，M=1 表示不区分。

随着版本的更新，VB 的字符处理机制也在变化。早期的 Windows 3.x 系统对字符采用 DBCS（Double Byte Character Set）编码方式。DBCS 编码实际上是一套混合编码，即西文采用 ASCII 编码，中文采用双字节编码。这种存储机制通常称为 ANSI 方式，其西文字符代码通常称为 ASCII 码，一个中文字符相当于由两个 ASCII 字符构成。早期版本的 VB 则使用 ANSI 编码方式（DOS 和 Windows 3.x 系统下），即一个西文字符用一个字节编码，中文字符用两个字节编码。这样，存储一个西文字符要占一个字节，而一个中文字符要占两个字节。

Windows 95 以后的版本采用 Unicode（国际化标准组织 ISO 字符标准）统一编码方式，即所谓大字符编码方案。这种方案把西文字符和中文字符作为整体统一编码，每个字符都用两个字节编码，在这种机制下，一个英文字符或一个汉字都被看做是一个字符，所占用的存储空间均为两个字节。VB 5.0/6.0 均采用 Unicode 编码方式来表示和存储字符串。

VB 中字符串的长度是以字为单位的，也就是说中、英文字符都具有一个“字”的长度，都占据两个字节的存储空间。但由于在 ANSI 方式中，对字符串的处理是以字节为单位的，所以在使用时要注意区分。

例如，用语句：

```
Print Len("欢迎使用 VB")
```

输出字符串的长度值，返回结果是 6。这与 ANSI 方式中的规定不符。为了解决这一问题，在 VB 中新增了一组字符串处理函数，以兼容原有的单字节处理方式（ANSI 方式）。这些新增的字符串函数与原有的相应函数相对应，只是在原函数名称后面加一个“B”字母，如 LenB, LeftB, RightB 等。例如：

```
Print LenB("欢迎使用 VB")
```

的返回结果就变成了 10。

VB 5.0 及其以后版本均采用 Unicode 编码方式来表示和存储字符串。Unicode 为了与传

统习惯码兼容，保留了 ASCII 码，只是将其字节数改为 2，增加的字节用 0 填充。为了实现这两种编码的转换，VB 提供了一个 StrConv 转换函数。格式为：

〈新字符串〉=StrConv(〈待转换字符串〉,〈转换格式〉)

其中，〈待转换字符串〉可以是字符串常量或变量，〈转换格式〉用来指定转换成哪种格式的字符串。与 ANSI 和 Unicode 转换有关的函数有：

- vbUnicode（64）
- 将 ANSI 编码格式的字符串转换成 Unicode 格式。
- vbFromUnicode（128）
- 将 Unicode 编码格式的字符串转换成系统的默认码页（DBCS 格式）。

2.5.3 日期和时间函数

日期和时间函数用来显示日期和时间，可提供某个事件何时发生及持续时间长短的信息。常用日期和时间函数见表 2-9。

表 2-9 常用日期和时间函数

函 数 名	说 明	示 例	结 果
Now	返回当前系统日期和时间（yy-mm-dd hh:mm:ss）	Now	11-09-26 10:36:06
Date()	返回当前系统日期（yy-mm-dd）	Date() 或 Date	11-09-26
Day(C   N)	返回日期值（1~31）	Day("11,09,26")	26
Month(C   N)	返回月份值（1~12）	Month("11,09,26")	9
MonthName(N)	返回月份中文名称	MonthName(9)	九月
Time()	返回系统时间	Time() 或 Time	10:36:08
Timer()	返回从午夜到现在经过的秒数	Timer() 或 Timer	55863.16
WeekDayName(N)	将星期值转换成星期名称（周日为 1）	WeekDayName(6)	星期五
Year(C   N)	返回年代号（1753~2078）	Year("11-09-26")	2011
DateSerial(年,月,日)	返回一个日期形式	DateSerial(11,9,26)	11-9-26
DateValue(C)	作用与 DateSerial()相同，只是 C 为字符串	DateValue("11,9,26")	11-9-26

【说明】

① 日期函数中，“C | N”参数表示自变量可以是字符表达式或数值表达式。若为数值表达式，表示距 1899 年 12 月 30 日前或后（正或负）的天数。例如，Day(10)结果为 9，表示 1899 年 12 月 30 日后 10 天是 9 号；Day(-10)结果为 20，表示 1899 年 12 月 30 日前 10 天是 20 号；year(8000)结果为 1921，表示 1899 年 12 月 30 日后 8000 天的年代号为 1921。

② 除了上面介绍的函数之外还有两个经常使用的函数：DateAdd 和 DateDiff。

DateAdd 函数返回结果包含一个日期，这一日期为加上了参数指定的时间间隔后的日期值。其语法格式为：

DateAdd(〈间隔单位〉,〈增减量〉,〈原日期变量〉)

例如，DateAdd("m",2,#8/26/2011#)表示 2011 年 8 月 26 日加两个月后的日期，输出结果为 11-10-26（2011 年 10 月 26 日）。关于间隔单位的规定见表 2-10。

表 2-10 常用日期函数间隔单位的规定

间隔单位	yyyy	q	m	y	d	w	ww	h	n	s
含 义	年	季	月	一年的日数	日	一周的日数	周	时	分	秒



可以使用 DateAdd 函数对日期加上或减去指定的时间间隔。例如，可以用 DateAdd 来计算距今天 30 天的日期，或者计算距现在 45 分钟的时间。

为了对日期加上“日”，间隔单位可以使用“一年的日数”(y)、“日”(d)或“一周的日数”(w)。

③ VB 还提供了一个建立日期的语句，格式为：

```
Date$=x$
```

其中 x\$可以是下列 4 种形式之一：

```
mm-dd-yy      mm-dd-yyyy      mm/dd/yy      mm/dd/yyyy
```

若用 1 位数表示月或日也可以，此时系统默认其前 1 位是 0；若仅用 2 位表示年，系统默认这一年为 19yy，07 代表 2007 年。此外，不论在月、日、年之间输入的是除号 (/) 还是连字符 (-)，系统最后给出的均是“-”。

④ VB 还提供了一个建立时间的语句，格式为：

```
Time$=x$
```

其中 x\$可以是下列 3 种形式之一：

```
hh              只设置小时，分和秒都默认为 0
hh:mm          只设置小时和分，秒默认为 0
hh:mm:ss       设置小时、分、秒
```

使用 Time\$语句时需注意，hh 的设置范围是 0~23。因此，若想设置为下午 1 时，必须把 hh 设置为 13。

### 2.5.4 格式输出函数

用格式输出函数 Format()可以使数值、日期或字符型数据按指定的格式输出。Format 函数的语法格式为：

```
Format(〈表达式〉,〈格式字符串〉)
```

【说明】

- ① 〈表达式〉可以是数值型、日期型或字符型的表达式。
  - ② 〈格式字符串〉是一个字符串常量或变量，由专门的格式说明字符组成。这些说明字符决定了数据项〈表达式〉的显示格式和长度。
  - ③ 当〈格式字符串〉是字符串常量时，必须放在双引号中。
  - ④ 格式输出函数 Format()返回一个 Variant 类型的值。
- 格式说明字符按照类型可以分为数值型说明符、日期型说明符和字符型说明符，其作用分别参见表 2-11、表 2-12 和表 2-13。

表 2-11 常用的数值型格式说明符

字 符	说 明	例 子
#	数字占位符。显示 1 位数字或什么都不显示。如果表达式在格式字符串中#的位置上有数字存在，那么就显示出来；否则，该位置就什么都不显示	Format(123.45,"####.###") 返回：123.45
0	数字占位符。显示 1 位数字或者 0。如果表达式在格式字符串中 0 的位置上有 1 位数字存在，那么就显示出来；否则，就以 0 显示	Format(123.45,"0000.000") 返回：0123.450

续表

字 符	说 明	例 子
.	小数点占位符	
,	千分位符号占位符	Format(1234.5, "#,###.##") 返回: 1,234.5
%	百分比符号占位符。表达式乘以 100，将百分比符号（%）插入到格式字符串中	Format(0.12345, "0.00%") 返回: 12.35%

表 2-12 常用的日期型格式说明符

字 符	说 明	例 子
dddddd	以完整日期表示法显示当前系统时间（包括年、月、日）	Format(Date, "dddddd") 返回: 2007 年 9 月 15 日
mmmm	以英文全称来表示月（January~December）	Format(Date, "mmmm") 返回: May
yyyy	以 4 位数来表示年	Format(Date, "yyyy") 返回: 2007
Hh	以有前导零的数字来显示小时（00~23）	Format(Time, "Hh:Nn:Ss") 返回: 08:06:58
Nn	以有前导零的数字来显示分（00~59）	
Ss	以有前导零的数字来显示秒（00~59）	
ttttt	以完整时间表示法显示（包括时、分、秒），用系统识别的时间格式定义的时间分隔符进行格式化。默认的时间格式为 hh:mm:ss	Format(Time, "ttttt") 返回: 20:57:06
AM/PM	在中午前以使用 AM 符号来表示，在中午以后用 PM 来表示	Format(Time, "tttttAM/PM") 返回: 20:57:46PM

表 2-13 常用的字符型格式说明符

字 符	说 明	例 子
@	字符占位符。显示字符或空白。如果字符串在格式字符串中@的位置有字符存在，那么就显示出来；否则，就在那个位置上显示空白。除非有惊叹号（!）在格式字符串中，否则字符占位符将由右向左被填充	Format("ABab", "@@@@@@") 返回: " ABab"
&	字符占位符。显示字符或什么都不显示。如果字符串在格式字符串中&的位置有字符存在，那么就显示出来；否则，就什么都不显示。除非有惊叹号（!）在格式字符串中，否则字符占位符将由右向左被填充	Format("ABab", "&&&&&&") 返回: "ABab"
<	强制小写。将所有字符以小写格式显示	Format("ABab", "<@@@@@") 返回: " abab"
>	强制大写。将所有字符以大写格式显示	Format("ABab", ">@@@@@") 返回: " ABAB"
!	强制由左向右填充字符占位符	Format("ABab", "!@@@@@") 返回: "ABab "

2.5.5 随机数语句和函数

在测试、模拟和游戏程序中，经常要使用随机数。随机数函数和语句见表 2-14。

表 2-14 随机数函数和语句

函数和语句	说 明	示 例
Randomize	产生随机数的种子	Randomize
Rnd[(〈算术表达式〉)]	函数返回小于 1 且大于或等于 0 的值	Rnd

【说明】若 Rnd 函数中〈算术表达式〉的值小于 0，每次都使用这个值作为随机数种子，将得到相同的随机数；若其值大于 0 或省略〈算术表达式〉，产生序列中的下一个随机数；若其值等于 0，产生最近生成的数（即产生与上次相同的数）。

在调用 Rnd 函数前使用 Randomize 语句，可以使该函数产生出不同序列的随机数。如果希望产生 A~B 之间的随机整数（包括 A 和 B），可通过下列语句实现：

```
Int((B-A+1) * Rnd + A)
```

2.5.6 数据类型转换函数

在 VB 中，一些数据类型可以自动转换，例如，由数字组成的字符串可自动转换为数值型，但是，多数类型不能自动转换，这就需要用类型转换函数来强制转换。转换函数见表 2-15。

表 2-15 数据类型转换函数

函 数	返 回 类 型	参 数 范 围
Cbool(C   N)	Boolean	任何有效的字符串或数值表达式
Cbyte(N)	Byte	0~255
Ccur(N)	Currency	-922 337 203 685 477.580 8~922 337 203 685 477.580 7
Cdate(D)	Date	任何有效的日期表达式
CDbl(N)	Double	负数：-1.79769313486232E308—4.94065645841247E-324 正数：4.94065645841247E-324~1.79769313486232E308
Cint(N)	Integer	-32 768~32 767，小数部分四舍五入
CLng(N)	Long	-2 147 483 648~2 147 483 647，小数部分四舍五入
CSng(N)	Single	负数：-3.402823E38—1.401298E-45 正数：1.401298E-45~3.402823E38
CStr(N)	String	依据参数返回 CStr
CVar(N)	Variant	若为数值，则范围与 Double 相同；若不为数值，则范围与 String 相同
CVErr(N)	Error	将实数转换成错误值

每个类型转换函数都可以强制将一个表达式转换成某种特定的数据类型，例如：

```
Area = CDbl(txtLength.Text * txtWidth.Text)
```

注意：如果传递给函数的参数超过转换目标数据类型的范围，将发生错误。例如，如果想把 Long 型数转换成 Integer 型数，那么，Long 型数必须在 Integer 数据类型的有效范围之内。若想了解 VB 正在使用哪种变量类型，可使用 VarType 函数。例如：

```
a=time:Print VarType(a) ' 返回值为 7 表示日期型
```

类型代码的说明见表 2-16。

表 2-16 常用数据类型的 VB 常数代码值和说明

VB 常数	代码值	说 明	VB 常数	代码值	说 明
vbEmpty	0	Empty（未初始化）	vbObject	9	对象
vbNull	1	Null（无有效数据）	vbError	10	错误值
vbInteger	2	整数	vbBoolean	11	逻辑值
vbLong	3	长整数	vbVariant	12	Variant（只与变体中数组一起使用）
vbSingle	4	单精度浮点数	vbDataObject	13	数据访问对象
vbDouble	5	双精度浮点数	vbDecimal	14	十进制值
vbCurrency	6	货币值	vbByte	17	位值
vbDate	7	日期	vbUserDefinedType	36	包含程序员定义类型的变量
vbString	8	字符串	vbArray	8192	数组

【说明】VarType 函数自身从不对 vbArray 返回值。VarType 总是要加上一些其他值来指出一个具体类型的数组。常数 vbVariant 只与 vbArray 一起返回，以表明 VarType 函数的参数是一个 Variant 类型的数组。例如，对一个整数数组的返回值是 vbInteger + vbArray 或 8194。如果一个对象有默认属性，则 VarType(object)返回对象默认属性的类型。

2.5.7 Shell 函数

在 VB 中不但提供了可调用的内部函数，还可以调用各种应用程序，也就是说，凡是能够在 DOS 或 Windows 下运行的可执行程序，都可以在 VB 中通过 Shell 函数调用。Shell 函数在语法格式如下：

Shell(Pathname[,Windowstyle])

其中，Pathname 为包括在两个双引号中的可执行程序的有效路径。

Windowstyle 是一个可选参数，表示在程序运行时窗口的样式。如果 Windowstyle 省略，则程序以具有焦点的最小化窗口来运行。

关于窗口样式设置参数见表 2-17。

表 2-17 Shell 函数 Windowstyle 参数

VB 常数	值	说 明
vbHide	0	窗口被隐藏，且焦点会移到隐式窗口
vbNormalFocus	1	窗口具有焦点，且还原到它原来的大小和位置
vbMinimizedFocus	2	窗口以一个具有焦点的图标来显示
vbMaximizedFocus	3	窗口是一个具有焦点的最大化窗口
vbNormalNoFocus	4	窗口被还原到最近使用的大小和位置，而当前活动的窗口仍然保持活动
vbMinimizedNoFocus	6	窗口以一个图标来显示，而当前活动的窗口仍然保持活动

需要说明的是，如果 Shell 函数成功地执行了所要执行的文件，它会返回程序的任务 ID。任务 ID 是一个唯一的数值，用来指明正在运行的程序。如果 Shell 函数不能打开指定的程序，则会产生一个错误。

注意，在默认情况下，Shell 函数是以异步方式来执行其他程序的。也就是说，用 Shell 启动的程序可能还没有完成执行过程，就已经执行到 Shell 函数之后的语句了。

# 2.6 语句

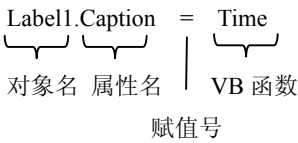
VB 中的语句是指执行具体操作的指令，每个语句行都以 Enter 键结束。

## 1. 程序语句

程序语句是 VB 关键字、属性、函数、运算符及能够生成 VB 编辑器可识别指令的符号的任意组合。一个完整的程序语句可以简单到只有一个关键字，例如：

Show

语句也可以是各种元素的组合，例如，下面的语句中，把当前系统时间赋值给 Label1 的 Caption 属性：



建立程序语句时必须遵从的构造规则称为语法。编写正确程序语句的前提，就是学习语言元素的语法，并在程序中使用这些元素正确地处理数据。

## 2. 语句的书写规则

在编写程序代码时要遵循一定的规则，这样写出的程序既能被 VB 正确地识别，又能增加程序的可读性。

### (1) 自动语法检查

在默认情况下，在输入语句的过程中，VB 将自动对输入的内容进行语法检查，如果发现语法错误，将弹出一个信息框提示出错的原因。如果没有进行自动语法检查，可执行“工具”→“选项”菜单命令，在“编辑器”选项卡中设置“自动语法检测”项。

### (2) 格式化处理

VB 会按约定对语句进行简单的格式化处理，例如，关键字或函数的第一个字母自动变为大写，在运算符前后加空格等。在输入语句时，关键字、函数等可以不必区分大小写。例如，在输入 Print 时，不管输入 Print, print, 还是 PRINT, 按 Enter 键后都变为 Print。为了提高程序的可读性，可在代码中应加上适当的空格，同时应按惯例处理字母的大小写。

### (3) 复合语句行

在一般情况下，输入程序时要求一行一句，一句一行。但是 VB 也允许使用复合语句行，即把几个语句放在一个语句行中，语句之间用冒号“:”隔开。一个语句行的长度最多不能超过 1 023 个字符。例如：

a = 2 : b = 3 : c = 4

### (4) 语句的续行

当一条语句很长时，在代码编辑窗口中阅读程序时不便查看，使用滚动条又比较麻烦。这时，就可以使用续行功能，用续行符“\_”（下划线）将一个较长的语句分为多个程序行。例如：

strMyStr="NAME : " & \_  
strName

在使用续行符时，在它前面至少要加一个空格，并且续行符只能出现在行尾。

3. 命令格式中的符号约定

为了便于解释，本书的语句、方法和函数格式中的符号采用统一约定。在各语句、方法、函数的语法格式和功能说明中，以尖括号“< >”、方括号“[ ]”、花括号“{ }”、竖线“|”、逗号加省略号“,...”、省略号“...”作为专用符号，这些符号的含义见表 2-18。

表 2-18 符号约定

符 号	含 义
< >	必选参数表示符。尖括号中为中文提示说明，实际使用时，要由使用者根据需要提供具体的参数。如果缺少必选参数，语句则发生语法错误
[ ]	可选参数表示符。方括号中的内容选与不选由程序员根据具体情况决定，且都不影响语句本身的功能。如果省略，则为默认值
	多中取一表示符，含义为“或者选择”。由竖线分隔的多个选择项，必须选择其中之一
{ }	包含多中取一的各项
,...	表示同类项目的重复出现
...	表示省略了在当时叙述中不涉及的部分

注意：这些专用符号和其中的中文提示，不是语句行或函数的组成部分。在输入具体命令或函数时，上面的符号均不可作为语句中的成分输入计算机，它们只是语句、函数格式的书面表示。例如：

```
[ <对象表达式> ] Print [ <表达式表> ] { , | ; }
```

习题 2

一、选择题

- 2.1 以下关于 VB 数据类型的说法，不恰当的是（ ）。
- A) VB 6.0 提供的数据类型主要有字符串型和数值型，此外还有字节、货币、对象、日期、布尔和变体数据类型等
- B) 目前 Decimal 数据类型只能在变体类型中使用
- C) 用户不能定义自己的数据类型
- D) 布尔型数据只能取两种值，用两个字节存储
- 2.2 以下各项，可以作为 VB 变量名的是（ ）。
- A) Book
- B) 2\_Seek
- C) 123.58
- D) Book-1
- 2.3 下列（ ）不能作为 VB 中的变量名。
- A) ABCDEFG
- B) P000000
- C) 89TWDDFF
- D) xyz
- 2.4 下列（ ）是 VB 中的合法变量名。
- A) AB7
- B) 7AB
- C) IF
- D) A[B]7
- 2.5 表达式  $2 * 3^2 + 2 * 8 / 4 + 3^2$  的值为（ ）。
- A) 64
- B) 31
- C) 49
- D) 22

2.6 函数  $\text{Int}(\text{Rnd}(0)*10)$  是在 ( ) 范围内的整数。

- A) (0,1)                      B) (1,10)                      C) (0,9)                      D) (1,9)

2.7 表达式  $3^2 \text{ Mod } 14 \setminus 2^3$  的值是 ( )。

- A) 1                      B) 0                      C) 2                      D) 3

2.8 在 VB 中, 下列两个变量名相同的是 ( )。

- A) Japan 和 Ja\_pan                      B) English 和 ENGLISH  
C) English 和 Engl                      D) China 和 Chin

2.9 数学式子  $\sin 25^\circ$  写成 VB 表达式是 ( )。

- A) Sin25                      B) Sin(25)  
C) Sin(25°)                      D) Sin(25\*3.14/180)

2.10 在 VB 中, 要强制用户对所用的变量进行显式声明, 可以在 ( ) 中设置。

- A) “属性”对话框                      B) “程序代码”窗口  
C) “选项”对话框                      D) 对象浏览器

2.11 下列符号常量的声明中, 不合法的是 ( )。

- A) Const a As Single = 1.1                      B) Const a = "OK"  
C) Const a As Double = Sin(1)                      D) Const a As Integer = "12"

2.12 在代码编辑器中, 续行符是换行书写同一个语句的符号, 用以表示续行符的是 ( )。

- A) 一个空格加一个下划线 “\_”                      B) 一个下划线 “\_”  
C) 一个连字符 “-”                      D) 一个空格加一个连字符 “-”

## 二、填空题

2.13 如果希望使用变量 x 来存放数据 765 432.123 456, 应将变量 x 声明为\_\_\_\_类型。

2.14 把 VB 算术表达式  $a/(b+c/(d+e/\text{Sqr}(f)))$  改写成数学表达式为\_\_\_\_\_。

2.15 如果 x 是一个正实数, 对 x 的第 3 位小数四舍五入的表达式是\_\_\_\_\_。

2.16 函数  $\text{Str}\$(256.36)$  的值是\_\_\_\_\_。

2.17 表达式  $(7 \setminus 2 + 1) * (8 \setminus 2 + 2)$  的值为\_\_\_\_\_。

2.18 下列语句的输出结果是\_\_\_\_\_。

Print Format \$(1258.6, "000,000.00")

## 三、思考题

2.19 VB 定义了哪几种数据类型? 变量有哪几种数据类型? 常量有哪几种数据类型?

2.20 下列数据哪些是变量? 哪些是常量? 是什么类型的常量?

- (1) name                      (2) "name"                      (3) False                      (4) ff  
(5) "11/16/99"                      (6) cj                      (7) "120"                      (8) n  
(9) #11/16/1999#                      (10) 12.345

2.21 VB 共有几种表达式? 根据什么确定表达式的类型?

2.22 在 VB 中, 对于没有赋值的变量, 系统默认值是什么?

2.23 将下列数学表达式改写为等价的 VB 算术表达式。

$$(1) \frac{1 + \frac{y}{x}}{1 - \frac{y}{x}}$$

$$(2) x^2 + \frac{3xy}{2-y}$$

$$(3) \sqrt{|ab-c^3|}$$

$$(4) \sqrt{s(s-a)(s-b)(s-c)}$$

2.24 写出下列表达式的值。

$$(1) (2 + 8 * 3) / 2$$

$$(2) 3^2 + 8$$

$$(3) \#11/22/99\# - 10$$

$$(4) "ZYX" \& 123 \& "ABC"$$

2.25 设 A = 7, B = 3, C = 4, 求下列表达式的值:

$$(1) A + 3 * C$$

$$(2) A^2 / 6$$

$$(3) A / 2 * 3 / 2$$

$$(4) A \text{ Mod } 3 + B^3 / C \setminus 5$$

2.26 写出下列表达式的值, 并在立即窗口中验证。

$$(1) "Visual" + "Basic"$$

$$(2) "xyz" \& 1234 \& "ABCD"$$

$$(3) "12345" < > "12345" \& "ABC"$$

$$(4) \text{Not } 2 * 5 < > 11$$

$$(5) 4 = 4 \text{ And } 5 > 2 + 2$$

$$(6) \text{Not } 8 < 5 \text{ Or } 9 > 3 \text{ And } 7 < 9 \text{ Or } 8 = 6$$

2.27 根据所给条件写出对应的逻辑表达式。

(1) 征兵的条件为: 男 (sex), 年龄 (age) 在 18~20 岁之间, 身高 (height) 在 1.65 米以上; 或者女, 年龄在 16~18 岁之间, 身高在 1.6 米以上。

(2) 工资调整的条件为: 工龄 (gongling) 在 15 年以上, 岗位 (gangwei) 是工人; 或者工龄在 10 年以上, 岗位是教师。

2.28 写出下列函数的值, 并在立即窗口中验证。

$$(1) \text{Int}(-3.14159)$$

$$(2) \text{Sqr}(\text{Sqr}(64))$$

$$(3) \text{Fix}(-3.1415926)$$

$$(4) \text{Int}(\text{Abs}(99 - 100) / 2)$$

$$(5) \text{Sgn}(7 * 3 + 2)$$

$$(6) \text{Month}("02,08,26")$$



## 第 3 章 Visual Basic 可视化编程的概念与方法

VB 采用的是面向对象、事件驱动编程机制，程序员只需编写响应用户动作的程序，如移动鼠标、单击等，而不必考虑按精确次序执行的每个步骤，编写代码相对较少。

本章主要介绍可视化编程的基本概念及设计步骤。

### 3.1 可视化编程的基本概念

VB 使用的可视化编程方法，是面向对象编程技术的简化版。在 VB 环境中所涉及的窗体、控件、部件和菜单项等均为对象，程序员不仅可以利用控件来创建对象，而且还可以建立自己的控件。

#### 3.1.1 对象

VB 具有面向对象程序设计的强大功能，程序的核心是对象（Object）。在 VB 中不仅提供了大量的控件对象，而且还提供了创建自定义对象的方法和工具，为开发应用程序带来了方便。

对象是现实生活中很常见的。可以把对象想象成日常生活中的各种物体，例如，一个人、一只气球、一辆汽车、一台微型计算机等都是对象。

以微型计算机为例，一台微机本身是一个对象，而微机又可以拆分为主板、CPU、内存、外设等部件，这些部件也是对象，因此，微机对象可以说是由多个“子”对象组成的，即一个容器（Container）对象。

与微机的概念类似，在 VB 程序中，窗体（Form）、命令按钮（Command Button）、列表框（List Box）等都是对象。

每个对象都有其特征（即属性），如小孩玩的气球，与它相关的特征数据有：直径、颜色、状态（充气或未充气）等，还有一些不可见的性质，如寿命等。当然，所有气球都具有这些属性，同时这些属性也会因气球的不同而不同。

#### 3.1.2 对象的属性、事件和方法

VB 的控件是具有自己的属性、事件和方法的对象，可以把属性看做一个对象的性质，把事件看做对象的响应，把方法看做对象的动作，由此构成了对象的三要素。

##### 1. 对象的属性

VB 程序中，每个对象都有用来描述和反映该对象特征参数，称做属性（Property），如：Name（控件名称）、Caption（标题）、Color（颜色）等。

在 VB 中，记录属性数据的地方叫做属性栏。属性栏中记录的属性数据叫做属性值。

在可视化编程中，每一种对象都有一组特定的属性。对象属性的设置一般有两种方式。

### (1) 预设法

在设计界面时，使用属性窗口设置对象的属性。这时只要在属性窗口中选中要修改的属性，然后在右列中输入新的值就可以了。

这种方法的特点是简单明了，每当选择一个属性时，在属性窗口的下部就显示该属性的一个简短提示。缺点是不能设置所有所需的属性。

### (2) 现改法

在编程中，通过程序代码更改对象的属性，这时可使用 VB 的赋值语句，格式为：

**对象名.属性名=属性值**

其中，“对象名.属性名”是 VB 中引用对象属性的方法。

下面语句可将标签对象 Label1 的 Caption 属性改为“结束”：

```
Label1.Caption="结束"
```

## 2. 对象的事件

### (1) 事件

对于对象而言，事件（Event）就是发生在该对象上的事情。例如，一个吹大的气球，用针扎它一下，该对象就会进行放气动作，“针扎”就是一个事件。

VB 中提供了许多对象，让用户利用它们来设计应用程序。例如，按钮就是一个对象。在按钮对象上最常发生的事就是“按一下”，这个“按一下”就是按钮对象的一个事件。在按钮上面用鼠标按一下，在 Windows 环境下称为“单击”，于是说，按钮会有有一个单击（Click）事件。

除了单击事件外，VB 中还有双击（DbClick）事件、装载（Load）事件、鼠标移动（MouseMove）事件等。不同的对象能够识别不同的事件，这就像老师可以批评学生，却不能去批评桌椅一样，因为桌椅不能识别“批评”这种事件的发生。

### (2) 事件过程

当在对象上发生了某个事件后，必须想办法处理这个事件，而处理的步骤就是事件过程（Event Procedure）。以气球为例，发生了“针扎”事件后，可能是进行粘补或丢弃，不论是粘补还是丢弃，都是针对“针扎”事件的处理步骤，也就是事件过程。

事件过程是针对事件而来的，而事件过程中的处理步骤在 VB 程序设计中就是所谓的程序代码。

在每一个 VB 提供的对象上面，都已经设定了该对象可能发生的事件，而每一个事件都会有一个对应的空事件过程。在编写程序时，并不需要把对象所有的事件过程填满，只要填入需要的部分就可以了。当对象发生了某一事件，而该事件所对应的事件过程中没有程序代码（也就是没有规定处理步骤）时，则表明程序对该事件“不予理会”。

### (3) 事件驱动

写完程序后开始执行时，程序会先等待某个事件的发生，然后再去执行处理此事件的事件过程。事件过程要经过事件的触发才会被执行，这种动作模式就称为事件驱动程序模式（Event Driven Programming Model），也就是说，由事件控制整个程序的执行流程。

### 3. 对象的方法

对象中除了属性之外，还包含了一些控制对象的动作或功能。以气球为例，假设气球这个对象有三个动作，分别是：充气（用氢气充满气球）、放气（排出气球中的气体）、上升（放手让气球飞走）。这三个动作都是气球这个对象所提供的功能，以程序设计术语来说，就是对象所提供的方法（Method）。

VB 的方法用于完成某种特定功能，如 Print（对象打印）方法、Show（显示窗体）方法、Move（移动）方法等。方法只能在代码中使用，其用法依赖于方法所需的参数个数及它是否具有返回值。当方法不需要参数并且也没有返回值时，可用下面的格式调用对象方法：

**对象名.方法名 [参数名表]**

例如，窗体 Form1 有输出方法 Print，在事件过程代码中下面语句可以输出文字内容“欢迎来到 Visual Basic 世界！”：

```
Form1.Print "欢迎来到 Visual Basic 世界！"
```

## 3.2 窗体、控件和代码窗口

在 VB 中，窗体、控件都是对象，用户通过代码窗口编写事件驱动程序。

### 3.2.1 窗体对象

窗体（Form）就是平时所说的窗口，是 VB 编程中最常见的对象，也是程序设计的基础。各种控件对象必须建立在窗体上，一个窗体对应一个窗体模块。

与 Windows 环境下的应用程序窗口一样，VB 中的窗体也具有控制菜单、标题栏、最大化/还原按钮、最小化按钮、关闭按钮和边框。

窗体的操作与 Windows 下的窗口操作一样，通过按住鼠标左键拖动标题栏可以移动窗体，鼠标对准窗体边框出现双向箭头时按住鼠标左键拖动可以改变窗体的大小。

窗体的控制菜单用来在程序运行时显示控制菜单，通过属性设置，可以在程序运行时将窗体上的标题栏隐藏起来。

### 3.2.2 控件

用 VB 开发程序就像盖房子一样，其中控件就像是盖房子用的钢筋、砖瓦等原材料。程序员使用不同的控件进行组合，并且设置内部的联系，就可以很方便地创建出程序来。

在 VB 中，控件是预先定义好的能够直接使用的对象，每个控件都有大量的属性、事件和方法，可在设计时或在代码中修改。

#### 1. 控件的画法

将工具箱中的控件添加到窗体中的过程称为“画控件”。画控件有两种方法：

- 单击工具箱中的控件按钮，在窗体上拖动鼠标画出控件，画出的控件大小和位置可随意确定；
- 双击工具箱中的控件按钮，在窗体的中央画出控件，画出的控件的大小和位置是暂时

固定的。

## 2. 控件的缩放和移动

在窗体上画出控件后，控件的边框上有 8 个蓝色小方块，这表明该控件是“活动”的，通常称为当前控件，如图 3-1 所示。单击控件，可以使之成为当前控件。

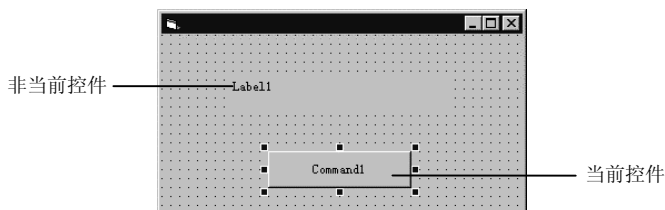



图 3-1 非当前控件和当前控件

对于选中的控件，可以用两种方法进行移动和缩放。

- 直接使用鼠标拖动控件到需要的地方。用鼠标指针对准控件的选中标志(8 个小方块)，出现双向箭头时，拖动鼠标可以改变控件的大小。
- 在属性窗口中修改某些属性来改变控件的位置和大小。与窗体和控件位置及大小有关的控件属性有：Left, Top, Width 及 Height。

## 3. 控件的复制与删除

在窗体上，控件的复制和删除操作与 Windows 环境下文件的操作相同。其操作步骤如下。

(1) 选中控件，单击工具栏上的“复制”按钮 ，将控件复制到剪贴板中。


(2) 单击“粘贴”按钮 ，将控件粘贴到窗体的左上角。由于复制控件名称相同，系统会提示是否创建控件数组，如图 3-2 所示。单击“否”按钮，在窗体上得到该控件的副本。副本的所有属性与原控件相同，只是名称属性 (Name) 的序号比原控件大。



图 3-2 询问是否创建控件数组

要删除活动控件，只需选中控件后按 Del 键，或者右键单击活动控件，在弹出的快捷菜单中选择“删除”命令即可。

## 4. 控件的布局

当窗体上存在多个控件时，需要对窗体上控件的排列方式、对齐方式、是否等大等格式进行调整。这些操作一般可以通过“格式”菜单完成。

要调整多个控件之间的位置，需要同时选定多个控件。常用的选定方法有两种：

- 在窗体的空白区域按住鼠标左键拖动画一个矩形框，将需要选中的控件圈上；
- 先按下 Shift 键，再用鼠标单击所要选中的控件。

在选定多个控件之后，就可以利用“格式”菜单对窗体上多个控件的格式进行调整了。

注意，当选择多个对象时，其中必有一个且只有一个是最后选择的对象，在这个对象的

边框上有 8 个实心小方块，其他被选对象的边框上则是 8 个空心小方块。多控件的格式操作都是以最后选择的对象为基准的。


### 3.2.3 代码窗口

#### 1. 代码窗口简介

代码窗口又称代码编辑器，各种通用过程和事件过程的代码均在此窗口上编写和修改。

##### (1) 打开代码窗口的方法

有下面 4 种方法可以打开代码窗口：

- 双击窗体的任何地方；
- 单击右键快捷菜单中的“查看代码”命令；
- 单击工程窗口中的“查看代码”按钮 ；
- 单击“视图”→“代码窗口”菜单命令。

##### (2) 代码窗口的组成



代码窗口如图 3-3 所示。

- “对象下拉列表框”中列出了当前窗体及所包含的全体对象名。其中，无论窗体的名称改为什么，作为窗体的对象名总是 Form。
- “过程下拉列表框”中列出了所选对象的所有事件名。
- “代码区”是程序代码编辑区，能够方便地进行代码的编辑和修改。另外，它还具有自动列出成员特性，能够自动列举属性值、方法或函数原型等性能，使代码编写更加方便。



图 3-3 代码窗口

##### (3) 代码查看视图

在代码窗口的左下角有两个按钮，即“过程查看”按钮  和“全模块查看”按钮 。这两个按钮可切换代码窗口的两种查看视图。

- “过程查看”按钮：一次只查看一个过程。
- “全模块查看”按钮：可查看程序中的所有过程。

#### 2. 自动功能

在 VB 代码窗口中编写代码时，VB 具有以下特性。

##### (1) 自动列出控件的属性和方法

当要输入控件的属性和方法时，在控件名后输入小数点，VB 就会自动显示出一个下拉列表框，其中包含了该控件的所有成员（属性和方法），如图 3-4 所示。输入属性名的前几个

字母，系统会自动检索并显示出需要的属性。

从列表中选中该属性名，按 Tab 键完成这次输入。当不熟悉控件有哪些属性时，这项功能是非常有用的。

如果系统设置为禁止自动列出成员，可使用快捷键 Ctrl+J 启用。

(2) 自动显示快速信息

该功能可显示语句和函数的语法格式。在输入合法的 VB 语句或函数名后，在当前行的下面自动显示该语句或函数的语法，如图 3-5 所示。

自动快速显示信息功能可以使用快捷键 Ctrl+I 启用。

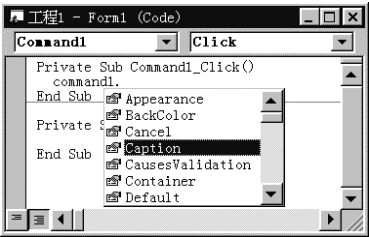


图 3-4 自动列出控件的属性和方法

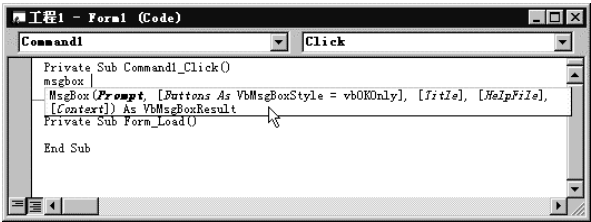


图 3-5 自动快速显示信息

(3) 自动语法检查

在 VB 中可自动进行语法检查。当输入某行代码后按 Enter 键，如果系统出现语法错误，VB 会显示警告提示框，同时该语句变成红色，如图 3-6 所示。



图 3-6 自动语法检查

### 3.3 可视化编程的一般步骤

VB 的对象已被抽象为窗体和控件，因而大大简化了程序设计。在用 VB 开发应用程序时，一般需要三个步骤：建立用户界面、设置属性和编写代码。

(1) 建立用户界面

用户界面由窗体和控件组成，所有控件都放在窗体上，程序中的所有信息都要通过窗体显示出来，它是应用程序的最终用户界面。在应用程序中要用到哪些控件，就在窗体上建立相应的控件。程序运行后，将在屏幕上显示由窗体和控件组成的用户界面。所以，要先建立窗体，然后在窗体上创建各种控件。

(2) 设置属性

建立界面后，就可以设置窗体和每个控件的属性。在实际的应用程序设计中，建立界面和设置属性可以同时进行，即每画完一个控件，接着就可以设置该控件的属性。当然，也可以在所有对象建立完成后再回来设置每个对象的属性。

### （3）编写代码

由于 VB 采用事件驱动编程机制，因此，大部分程序都是针对窗体中各个控件所能支持的方法或事件编写的，这样的程序称为事件过程。例如，命令按钮可以接收鼠标事件，如果单击该按钮，鼠标事件就调用相应的事件过程来做出相应的反应。

在具体的设计过程，也可以在创建对象的同时，一边设置对象的属性，一边编写事件过程代码。

下面以设计图 3-7 所示的“加法计算器”为例，讲述可视化编程的一般步骤。

#### 1. 新建一个工程

在 VB 中，开发的每个应用程序都被称为工程。新建一个工程有两种方法。

- 启动 VB 后，系统显示“新建工程”对话框，在“新建”选项卡中选择“标准 EXE”项，然后单击“打开”按钮。
  - 执行“文件”→“新建工程”菜单命令，在“新建工程”对话框中双击“标准 EXE”项。
- 进入 VB 的集成开发环境，开始设计工程，即应用程序。系统默认的窗体只有一个 Form1。

#### 2. 添加控件

向窗体中添加控件的步骤如下。

（1）单击工具箱中的“控件”图标，鼠标指针变成十字形状。

（2）在窗体的工作区按下鼠标左键拖动，即可在窗体上画出对应的控件。

在窗体 Form1 上绘出程序所需的控件，依次分别为文本框控件 Text1~Text3，标签控件 Label1，命令按钮控件 Command1，如图 3-8 所示，同类型控件的序号依次自动增加。

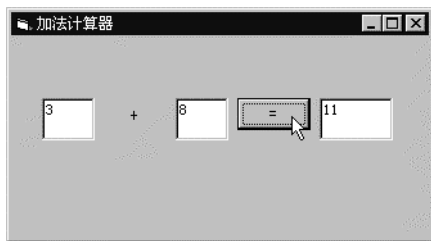


图 3-7 “加法计算器”示例

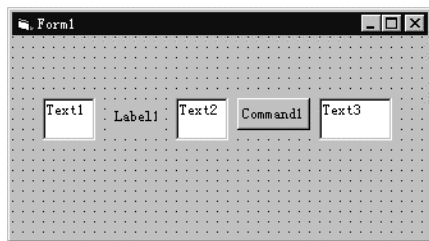


图 3-8 增加控件进行界面设计

#### 3. 设置属性

对象属性的设置是在属性窗口中进行的，其操作方法如下。

（1）设置窗体 Form1 的属性

单击窗体的空白区域（不要单击任何控件），确认选中的是窗体，可从图 3-9 右侧的对象下拉列表框中查看。

在属性窗口中找到标题属性 Caption，将其值改为“加法计算器”，如图 3-9 所示。

（2）设置控件的属性

单击窗体上的控件，确认选中该控件，根据需要逐一设置控件的各属性。

分别选中文本框控件 Text1~Text3，将其 Text 属性设置为空。

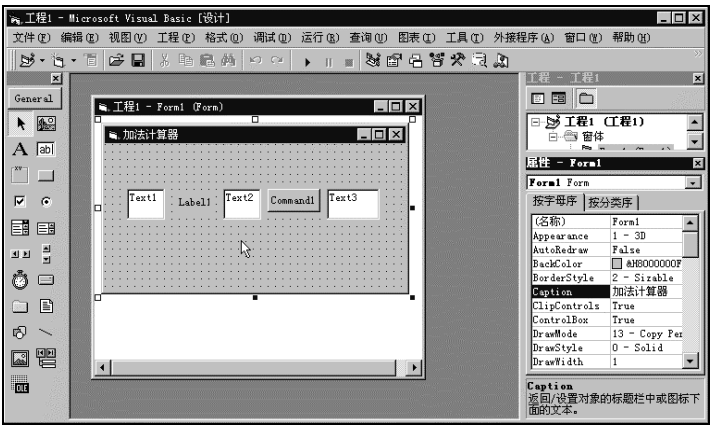


图 3-9 设置窗体 Form1 的属性

选中标签控件 Label1，将其 Caption 属性设为 “+”，将其 Alignment 属性改为 “2 - Center”，使其居中显示。

将命令按钮 Command1 的 Caption 属性设为 “=”。

属性设置后的窗体如图 3-10 所示。

4. 编写代码

在图 3-10 所示的窗体中，双击 “=” 按钮，打开代码编辑器。其中，事件过程的首尾两行是系统自动给出的代码，不必重复输入，例如：

```
Private Sub Command1_Click()  
.....  
End Sub
```

在首、尾两行代码之间输入命令按钮 Command1 的 Click（单击）事件过程代码，如图 3-11 所示。

```
Private Sub Command1_Click()  
    Text3.Text = Val(Text1.Text) + Val(Text2.Text)  
End Sub
```

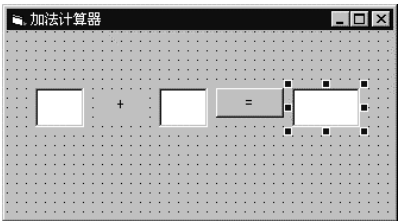


图 3-10 属性设置后的窗体

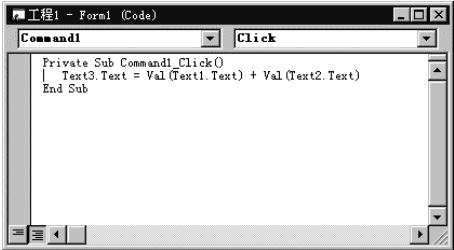


图 3-11 在代码窗口输入事件过程代码

5. 运行工程

单击工具栏上的“启动”按钮 ，如图 3-12 所示，运行工程。



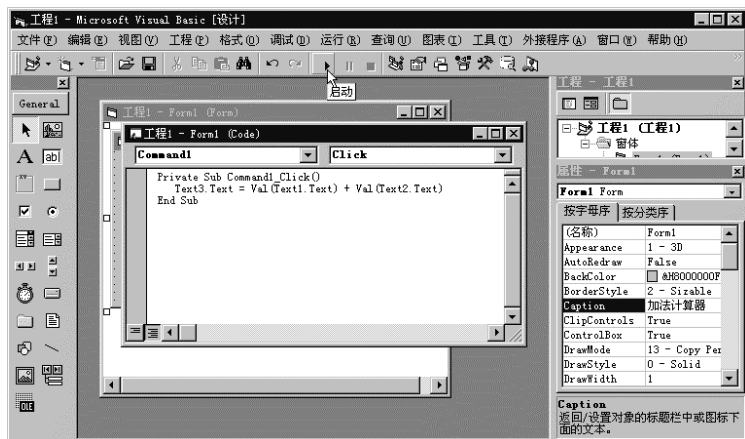




图 3-12 运行工程

用鼠标分别单击文本框 Text1, Text2, 输入数字, 单击 “=” 按钮, 则将在文本框 Text3 中显示结果, 如图 3-7 所示。


单击窗体标题栏上的“关闭”按钮  可关闭该窗口, 结束运行, 单击工具栏上的“结束”按钮  也可结束程序运行, 返回窗体设计器。

## 6. 修改工程

修改工程包括修改对象的属性和代码, 或者添加新的对象和代码, 或者调整控件的大小等, 直到满足工程设计需要为止。


## 7. 保存工程

在调试正确后需要保存工程, 即以文件的方式保存到磁盘上。常用下面两种方法保存工程:

- 执行“文件”→“保存工程”或“工程另存为”菜单命令。
- 单击工具栏上的“保存工程”按钮 .

如果新建工程从未保存过, 系统将打开“文件另存为”对话框, 如图 3-13 所示。

由于一个工程可能含有多种文件, 如工程文件和窗体文件等, 这些文件集合在一起才能构成应用程序。因此, 在“文件另存为”对话框中, 需注意保存类型, 并且将窗体文件 (\*.frm) 保存到指定文件夹中。窗体文件存盘后系统会弹出“工程另存为”对话框, 保存类型为“工程文件 (\*.vbp)”, 默认工程文件名为“工程 1.vbp”, 保存工程文件到指定文件夹中。建议将同一工程所有类型的文件存放在同一文件夹中。

如果想保存修改后磁盘上已有的工程文件, 可直接单击工具栏上的“保存工程”按钮 , 这时系统不会弹出“文件另存为”对话框。

## 8. 工程的编译

当完成工程的全部文件之后, 可将此工程转换成可执行文件 (.exe), 即编译工程。

在 VB 中, 对程序 (工程) 的编译操作非常简单。首先在“文件”菜单中选择“生成工程 1.exe”命令, 在打开的“生成工程”对话框中选择程序要存放的文件夹和文件名, 如图 3-14 所示, 单击“确定”按钮即可生成 Windows 中的应用程序。



图 3-13 保存工程



图 3-14 编译工程

## 习题 3

### 一、选择题

3.1 在 VB 中，被称为对象的是（ ）。

- A) 窗体                      B) 控件                      C) 控件和窗体                      D) 窗体、控件和属性

3.2 关于 VB “方法”的概念错误的是（ ）。

- A) 方法是对象的一部分                      B) 方法是预先定义好的操作  
C) 方法是对事件的响应                      D) 方法用于完成某些特定的功能

3.3 VB 程序设计采用的编程机制是（ ）。

- A) 可视化                      B) 面向对象                      C) 事件驱动                      D) 过程结构化

3.4 确定窗体控件启动位置的属性是（ ）。

- A) Width 和 Height                      B) Width 或 Height  
C) StartUpPosition                      D) Top 和 Left

3.5 下列说法正确的是（ ）。

- A) 对象的可见性可设为 True 或 False  
B) 标题的属性值不可设为任何文本  
C) 属性窗口中属性只能按字母顺序排列  
D) 某些属性的值可以跳过不设置，自动设为空值

### 二、填空题

3.6 VB 对象可以分为两类，分别为\_\_\_\_和\_\_\_\_。

### 三、思考题

3.7 试述可视化编程中对象、属性、事件和方法的含义。

3.8 简述 VB 可视化编程的一般步骤。

## 第 4 章 顺序结构程序设计

程序界面设计完成后，就可以编写事件代码来驱动程序运行了，编写的代码称为计算机程序。一个计算机程序通常分为输入、处理、输出三部分。计算机通过输入操作接收数据，然后对数据进行处理，并将处理完的数据以完整有效的方式提供给用户。

VB 的输入/输出操作有着十分丰富的内容和形式。VB 中提供了多种手段，并可通过各种控件实现输入/输出操作，使输入/输出更加灵活多样、方便直观。

### 4.1 顺序结构程序的概念

VB 虽然采用事件驱动方式调用相对划分得比较小的子过程，但是对于具体的过程本身，仍然要用到结构化程序的方法，用控制结构控制程序执行的流程。有些简单程序可以只用单向的顺序流程来编写，有些流程可以依靠运算符的优先级来控制，但为了处理复杂问题，仍要通过选择和循环改变语句执行的顺序。结构化程序设计有三种基本结构：顺序结构、选择结构和循环结构。

如果没有使用控制流程语句，程序便从左至右、自顶向下地顺序执行这些语句，即程序为顺序结构。顺序结构是一种线性结构，也是程序设计中最简单、最常用的基本结构。其特点是：在该结构中，各语句按照各自出现的先后顺序依次执行。一个程序通常可分为三个部分：输入、处理和输出。

用程序处理实际问题时，常常需要用户输入数据或要求程序输出数据。下面介绍几个语句，并由它们组成顺序结构。这些语句包括数据输出语句、赋值语句，以及标签、文本框、对话框等控件。

### 4.2 数据输出

一个没有输出操作的程序是没有什么实用价值的。VB 的输出操作包括文本信息的输出和图形图像的输出，本章主要介绍文本信息的输出。

#### 4.2.1 直接输出到窗体

##### 1. 使用 Print 方法

在 VB 中，可以使用 Print 方法实现数据输出。Print 方法可以在窗体上输出文本字符串或表达式的值，并可在其他图形对象或打印机上输出信息。其语法格式为：

**[〈对象名〉.] Print [〈表达式〉] [{,|;}]**

##### 【说明】

① 如果使用 Print 方法将数据输出到窗体上，应先使用 Show（显示）方法，否则输出

数据不可见。

② 格式中的〈对象名〉可以是 Form（窗体）、PictureBox（图片框）或 Printer（打印机）。如果省略，则在当前窗体上直接输出。

例如，直接将字符串“你好！”输出到当前窗体上，代码如下：

```
Show
Print "你好！"
```

又如，将字符串“Hello”在 Picture1（图片框）上显示出来，代码如下：

```
Picture1.Print "Hello"
```

再如，将字符串“Good morning”输出到 Printer（打印机），代码如下：

```
Printer.Print "Good morning"
```

③ 〈表达式表〉由一个或多个表达式组成，可以是数值表达式或字符串。对于数值表达式，将输出表达式的值；对于字符串，则照原样输出。如果省略〈表达式表〉，则输出一个空行。

```
Show
x = 1 : y = 2
Print x                ' 输出变量 x 的值
Print                  ' 输出空行
Print "Hello"          ' 字符串必须放在双引号内
```

输出结果为：

```
1
```

```
Hello
```

输出数据时，数值数据的前面有一个符号位，后面有一个空格，而字符串前后都没有空格。

④ 当输出多个表达式时，各表达式之间用分隔符逗号“,”或分号“;”隔开。

- 如果使用逗号分隔符，则各输出项按标准输出（分区输出）格式显示，此时，以 14 个字符宽度为单位将输出行分为若干区段，逗号后面的表达式在下一个区段输出。
- 如果使用分号分隔符，则按紧凑格式输出，即数值型数据后多一个空格，字符串后没有空格。

例如：

```
a = 2 : b = 4 : c = 6
Show
Print a, b, c, "stud", "ent"
Print -2; -4; "stud"; "ent"; -6
```

输出结果为：

```
2          4          6          stud          ent
-2 -4 student-6
```

⑤ 如果在语句行的末尾使用逗号分隔符，则下一个 Print 输出的内容将在当前 Print 所输出信息的下一个分区显示；如果在语句行的末尾使用分号分隔符，则下一个 Print 输出的内容将紧跟在当前 Print 所输出的信息后面；如果省略语句行末尾的分隔符，则 Print 方法将自动换行。

⑥ Print 方法具有计算和输出的双重功能，对于表达式，总是先计算后输出。例如：

```
x=2 : y=3
Print (x+y)*2
```

该例中的 Print 方法先计算表达式(x+y)\*2，然后输出其值 10。

【例 4-1】 用 Print 方法输出数据到窗体上，程序的执行结果如图 4-1 所示。  
设计步骤如下。

- (1) 建立应用程序用户界面。新建一个工程，进入窗体设计器，如图 4-2 所示。
- (2) 设置对象属性。由于不需要在窗体上放置控件，所以这里不需要设置对象属性。
- (3) 编写事件代码。由于要求在运行程序时直接显示输出结果，所以使用窗体 Form 的 Load 事件。右键单击窗体，打开快捷菜单，如图 4-3 所示。从快捷菜单中单击“查看代码”项，打开代码窗口，如图 4-4 所示。从对象下拉列表框中选中“Form”项，从过程下拉列表框中选中“Load”项，如图 4-5 所示。在代码区中输入 Form\_Load()的代码，如图 4-6 所示。

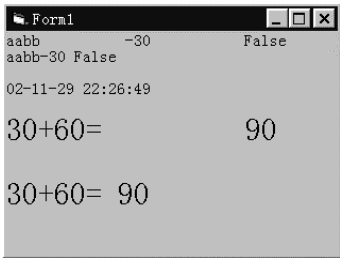


图 4-1 程序的执行结果

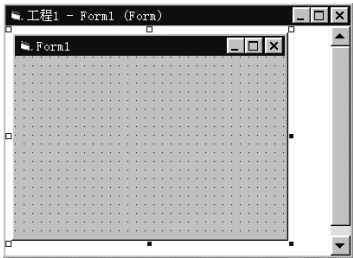


图 4-2 新建窗体设计器

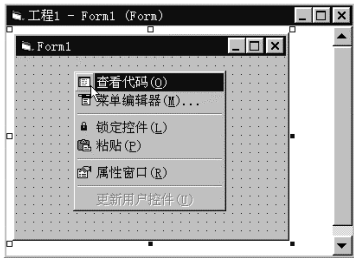


图 4-3 窗体的快捷菜单



图 4-4 代码窗口

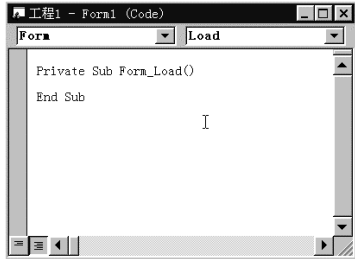


图 4-5 选取对象和过程

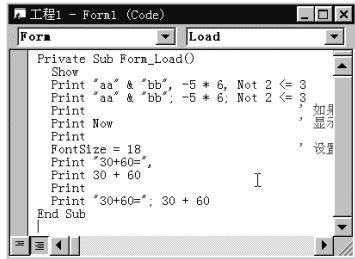



图 4-6 输入代码

窗体 Form 的 Load 事件代码如下：

```
Private Sub Form_Load()
    Show
    Print "aa" & "bb", -5 * 6, Not 2 <= 3
    Print "aa" & "bb"; -5 * 6; Not 2 <= 3
    Print ' 如果 Print 后不用任何项，则输出一个空行
    Print Now ' 显示当前日期和时间
    Print
    FontSize = 18 ' 设置字体尺寸
    Print "30+60=",
    Print 30 + 60
```

```
Print
Print "30+60="; 30 + 60
```

**End Sub**

单击工具栏中的“启动”按钮执行程序，运行结果如图 4-1 所示。执行“文件”→“工程另存为”菜单命令保存工程。程序调试完成后，可执行“文件”→“移除工程”菜单命令，结束本次程序的设计。

**【例 4-2】** 使用 Print 方法在窗体上直接输出字符串或数值表达式的值。

设计步骤如下。


(1) 建立应用程序用户界面。新建一个工程，进入窗体设计器，在窗体中增加一个命令按钮 Command1，如图 4-7 所示。

(2) 设置对象属性。设置 Command1 的 Caption 属性为“欢迎”。

(3) 编写事件代码。编写“欢迎”命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()
    Print
    Print "2 * 3 + 4 ="; 2 * 3 + 4           ' 使用“;”分隔符
    Print                                     ' 输出一个空行
    Print "祝您学好"
    Print , "Visual"                         ' 使用“,”分隔符
    Print , , "Basic"                       ' 使用两个“,”分隔符
    Print
    Print "    祝您学好",                   ' 在行末使用“,”分隔符
    Print "Visual"; " Basic"
```

**End Sub**

单击工具栏中的“启动”按钮执行程序，首先显示如图 4-8 (a) 所示的窗口，单击“欢迎”按钮，将显示如图 4-8 (b) 所示的窗口。

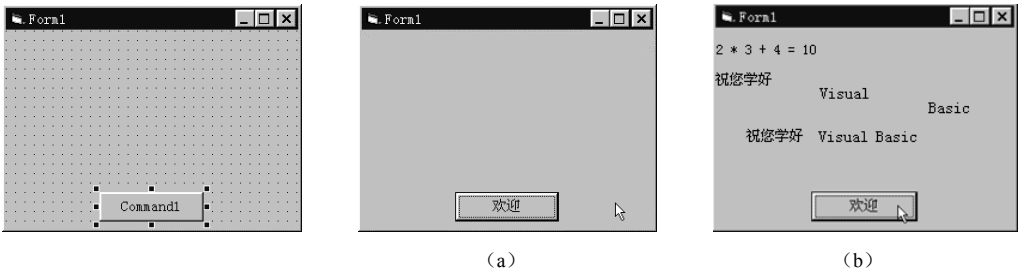


图 4-7 增加一个命令按钮

图 4-8 运行程序

2. 与 Print 方法有关的函数

为了使数据按指定的格式输出，VB 提供了 Tab, Spc 等函数，这些函数可以与 Print 方法配合使用。

(1) Tab 函数

在 Print 方法中，可以使用 Tab 函数对输出进行定位，其格式为：

**Tab(n);**

**【说明】**

① n 为数值表达式，其值为一个整数。Tab 函数把显示或打印位置移到由参数 n 指定的列，从此列开始输出数据。要输出的内容放在 Tab 函数后面，并用分号隔开。例如：

```
Print Tab(20); "姓名"; Tab(40); "班级"; Tab(60); "年龄"
```

② 通常最左边的列号为 1。如果当前的显示位置已经超过 n，则自动下移一行；当 n 大于行的宽度时，显示位置为：

```
n Mod <行宽>
```

③ 当在一个 Print 方法中有多个 Tab 函数时，每个 Tab 函数对应一个输出项，各输出项之间用分号隔开。

**【例 4-3】** 使用 Tab 函数对齐输出，如图 4-9 所示。

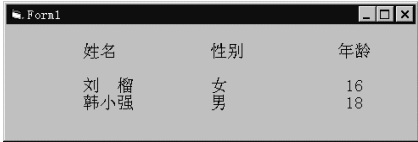


图 4-9 使用 Tab 函数对齐输出

编写窗体 Form 的 Click 事件代码：

```
Private Sub Form_Click()  
    Show  
    FontSize = 12 ' 设置输出文本字体的大小  
    Print  
    Print Tab(10); "姓名"; Tab(25); "性别"; Tab(40); "年龄"  
    Print  
    Print Tab(10); "刘 榴"; Tab(25); "女"; Tab(40); 16  
    Print Tab(10); "韩小强"; Tab(25); "男"; Tab(40); 18  
End Sub
```

**(2) Spc 函数**

在 Print 方法中，也可以使用 Spc 函数来对输出进行定位。与 Tab 函数不同，Spc 函数提供若干空格，其格式为：

**Spc(n);**

**【说明】**

① n 为数值表达式，其值为一个整数，表示在显示或打印下一个表达式之前插入的空格数。

② Spc 函数与输出项之间用分号隔开。例如：

```
Print "姓名"; Spc(5); "性别"; Spc(5); "年龄"
```

③ 当 Print 方法与不同大小的字体一起使用时，使用 Spc 函数打印的空格字符的宽度总是等于选用字体内以磅数为单位的所有字符的平均宽度。

④ Spc 函数与 Tab 函数的作用类似，可以互相代替。但应注意，Tab 函数从对象的左端开始计数，而 Spc 函数只表示两个输出项之间的间隔。

3. 使用位置属性和字体属性

要精确地把文本输出到窗体、图片框或打印页上，可以使用位置属性 `CurrentX` 和 `CurrentY`。这两个属性分别表示当前输出位置的横坐标与纵坐标。

如果要控制所显示或打印文本的大小和外观，可以用 VB 中的字体属性。各字体属性及其名称见表 4-1。

表 4-1 字体属性及其名称

属 性	名 称	属 性	名 称
FontName	字体名	FontSize	字号大小
FontBold	字体样式粗体	FontStrikethru	加删除线
FontItalic	字体样式斜体	FontUnderline	加下划线

另外，还可以在 `Print` 方法中用 `Format` 函数格式化输出格式。

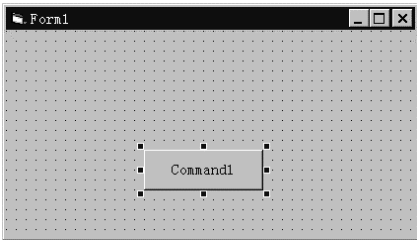


图 4-10 `CurrentX` 和 `CurrentY` 示例

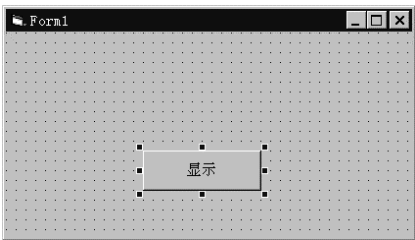
【例 4-4】 如图 4-10 所示，按某字体大小，把字符串“五光十色”输出到窗体上，坐标位置为 (800,600)。设计步骤如下。

(1) 建立用户界面。新建一个工程，进入窗体设计器，增加一个命令按钮 `Command1`，如图 4-11 (a) 所示。

(2) 设置对象属性。将命令按钮 `Command1` 的 `Caption` (标题) 属性改为“显示”，设置属性后的界面如图 4-11 (b) 所示。



(a)



(b)

图 4-11 建立用户界面并设置对象属性

(3) 编写事件代码。

“显示”命令按钮 `Command1` 的 `Click` 事件代码如下：

```
Private Sub Command1_Click()  
    Dim a As String  
    FontName = "隶书"           ' 设置输出文本的字体  
    FontSize = 18               ' 设置输出文本的字号  
    a = "五光十色"  
    CurrentX = 800              ' 设置输出的水平位置  
    CurrentY = 600              ' 设置输出的垂直位置  
    Show  
    Print a                     ' 输出文本  
End Sub
```



运行程序，结果如图 4-10 所示。

#### 4. 清除方法 Cls

使用 Cls 方法可以清除 Form（窗体）或 PictureBox（图片框）中由 Print 方法或图形方法在运行时所生成的文本或图形，清除后的区域以背景色填充。Cls 方法的语法格式为：

[〈对象名〉.]Cls

**【说明】**

① 〈对象名〉可以是 Form 或 PictureBox。如果省略〈对象名〉，则清除窗体上由 Print 方法或图形方法在运行时所生成的文本或图形。

② 设计时使用 Picture 属性设置的背景位图和放置的控件不受 Cls 方法影响。

**【例 4-5】** 使用 Cls 方法清除例 4-4 窗体中由 Print 方法所生成的文本，如图 4-12 所示。

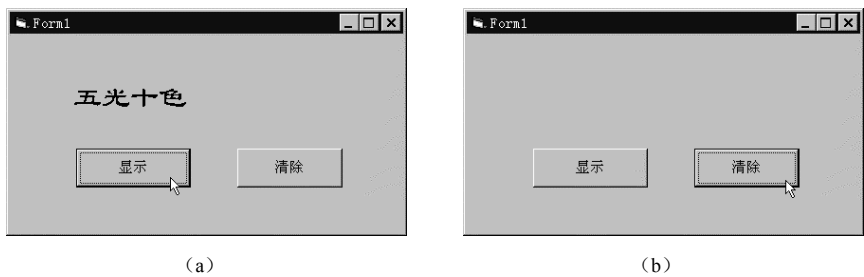


图 4-12 Cls 方法示例

只需在例 4-4 代码中增加一个命令按钮 Command2，将 Caption 属性改为“清除”，并编写其 Click 事件代码：

```
Private Sub Command2_Click()  
    Cls  
End Sub
```

#### 4.2.2 使用标签控件输出

标签（Label）主要是用来显示（输出）文本信息的，不能作为输入信息的界面，也就是说，标签控件的内容只能用 Caption 属性来设置或修改，不能直接编辑。它是 VB 中最常用的输出文本信息的工具，完全可以取代 Print 方法。

##### 1. 标签控件的常用属性

（1）Caption 属性

该属性用于在标签中显示文本。在默认情况下，Caption 是 Label 控件中唯一的可见部分。

（2）BorderStyle（边框样式）属性

该属性用来设置标签的边框。该属性可以取两个值，即 0 或 1。在默认情况下，该属性值为 0，标签无边框。如果把 BorderStyle 属性设置成 1，那么标签就有了一个边框。

（3）其他外观属性

可以通过设置标签的 BackColor（背景色）、ForeColor（前景色）和 Font（字体）等属性来改变标签的外观。

2. 标签使用示例

【例 4-6】 如图 4-13 所示，使标签控件具有边框，然后再修改标签属性为无边框。

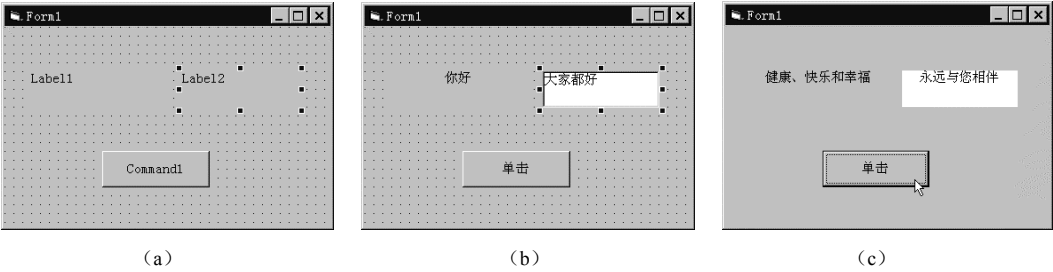


图 4-13 有边框和无边框的标签控件

设计步骤如下。

- (1) 建立用户界面。新建一个工程，进入窗体设计器，增加一个命令按钮 Command1、两个标签 Label1 和 Label2，如图 4-13 (a) 所示。
- (2) 设置对象属性。分别设置对象的属性，见表 4-2。设置后的界面，如图 4-13 (b) 所示。

表 4-2 属性设置

对 象	属 性	属 性 值	说 明
Command1	Caption	单击	按钮的标题
Label1	Caption	你好	标签的内容
	Alignment	2-Center	标签的内容居中显示
Label2	Caption	大家都好	标签的内容
	BorderStyle	1-Fixed Single	有边框的标签
	BackColor	&H00FFF	标签的背景色改为白色

- (3) 编写事件代码。“单击”命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()  
    Label1.Caption = "健康、快乐和幸福" ' 改变 Label1 的标题内容  
    Label2.Caption = "永远与您相伴" ' 改变 Label2 的标题内容  
    Label2.Alignment = 2 ' Label2 的内容居中显示  
    Label2.BorderStyle = 0 ' 将 Label2 的边框样式改为无边框  
End Sub
```

运行程序，单击“单击”按钮，结果如图 4-13 (c) 所示。

4.3 常用基本语句

VB 中常用的基本语句有赋值语句 Let、卸载对象语句 Unload 和注释语句 Rem 等。

4.3.1 赋值语句 Let

赋值语句是任何程序设计中最基本的语句。在前面的例子中，已经在代码中使用了它。它的作用是将指定的值赋给某个变量或对象的某个属性。

赋值语句的语法格式为：

[Let] 〈名称〉 = 〈表达式〉

【说明】

- ① Let 表示赋值，通常省略。
- ② 〈名称〉是变量或属性的名称。
- ③ 〈表达式〉可以是算术表达式、字符串表达式、关系型表达式或逻辑表达式，其类型应与变量名的类型一致，即同时为数值型或同时为字符型，否则会出现“类型不匹配”的错误。当同时为数值型但有不同的精度时，强制转换成“=”左边的精度。
- ④ 赋值语句先计算〈表达式〉，然后再赋值。
- ⑤ 格式中的赋值号不是数学上的等号。例如，语句 a = 2 应读做“将数值 2 赋给变量 a”或是“使变量 a 的值等于 2”，可以理解为： $a \leftarrow 2$ 。虽然赋值号与关系运算符的等号都用“=”表示，但 VB 系统不会产生混淆，它将根据其所处的位置自动判断是哪种意义的符号。

【例 4-7】 设计程序，交换两变量的值，如图 4-14 所示。



图 4-14 交换两变量的值

【分析】将两个不同的变量假设为两个瓶子 A 和 B，其中分别装有不同颜色的液体，现在需要交换瓶子中的液体。可以这样来做：另取一个瓶子 C，先将瓶 A 中的液体倒入瓶 C 中，再将瓶 B 中的液体倒入瓶 A 中，最后将瓶 C 中的液体倒入瓶 B 中。

设计步骤如下。

- (1) 建立用户界面。新建一个工程，进入窗体设计器，增加一个命令按钮 Command1、4 个标签 Label1~Label4，如图 4-15 (a) 所示。
- (2) 设置对象属性，见表 4-3。设置属性后的界面如图 4-15 (b) 所示。

表 4-3 设置属性

对 象	属 性	属 性 值	说 明
Command1	Caption	交换两变量的值	按钮的标题
Label1	Caption	A 的值为	标签的内容
Label2	Caption	2	标签的内容
	BorderStyle	1-Fixed Single	有边框的标签
Label3	Caption	B 的值为	标签的内容
Label4	Caption	3	标签的内容
	BorderStyle	1-Fixed Single	有边框的标签

(3) 编写事件代码。“交换两变量的值”命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()
```

```
c = Label2.Caption
Label2.Caption = Label4.Caption
Label4.Caption = c

End Sub
```

运行程序，结果如图 4-14 所示。

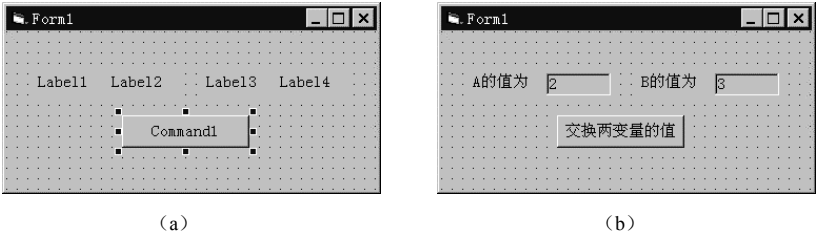


图 4-15 建立界面与设置属性

4.3.2 卸载对象语句 Unload

当要结束应用程序，或从内存中卸载窗体，或从内存中卸载某些控件时，可以使用 Unload 语句。Unload 语句的语法格式为：

```
Unload <对象名>
```

【说明】

- ① <对象名> 是要卸载的窗体对象或控件的名称，可以用 Me 来表示当前所在的窗体对象。
- ② 在卸载窗体前，会发生 QueryUnload（窗体队列关闭）事件，然后是 Unload（卸载）事件。在其中任一事件过程代码中设置 Cancel 参数为 True 可防止窗体被卸载。

【例 4-8】 在例 4-7 窗体上增加“关闭”命令按钮，关闭窗体。程序界面如图 4-16 所示。

设计步骤如下。

只需在例 4-7 窗体上增加一个命令按钮 Command2，将其 Caption 属性改为“关闭”，并编写 Command2 的 Click 事件代码如下：

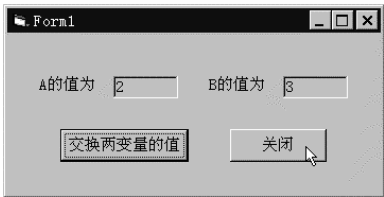


图 4-16 关闭窗口

```
Private Sub Command2_Click()
    Unload Me
End Sub
```

' Me 表示按钮所在的窗体对象

运行程序，结果如图 4-16 所示。

4.3.3 注释语句 Rem

为了提高程序的可读性，通常应在程序的适当位置加上一些注释。注释语句用来在程序中包含注释，语法格式为：

```
Rem <注释内容>
```

或

```
' <注释内容>
```

### 【说明】

①〈注释内容〉指要包含的任何注释文本。在 **Rem** 关键字与注释内容之间要加一个空格。可以用一个英文单引号 “'” 来代替 **Rem** 关键字。

② 如果在其他语句行后使用 **Rem** 关键字，必须用冒号 “:” 与前面语句隔开。若使用英文单引号，则在其他语句行后不必加冒号。

例如，

```
c = Label2.Caption           'c 为临时变量
Label2.Caption = Label4.Caption : Rem 将 Label4 的 Caption 属性值赋给 Label2
```

## 4.4 利用文本框输入数据

如果程序没有输入操作，必然缺乏灵活性。在 VB 中，允许用户输入文本信息的最直接的方法是使用文本框。

### 4.4.1 文本框控件

文本框 (TextBox) 是一个文本编辑区域，用户可以在该区域中输入、编辑和显示文本内容。

#### 1. 文本框控件的常用属性

文本框的常用属性有以下 6 个。

##### (1) Text 属性

Text 属性表示文本框中包含的文本内容。

##### (2) Locked 属性

Locked 属性决定控件是否可编辑。当 Locked 属性的值为 True 时，文本框的内容不可编辑；为 False 时，可编辑。默认为 False。

##### (3) MultiLine (多行) 属性

若将控件的 MultiLine 属性设置为 True，则可以输入多行文本，并且文本的内容可多达 32KB。默认为 False。

##### (4) ScrollBars (滚动条) 属性

ScrollBars 属性决定文本框中是否显示滚动条及滚动条的显示形式。默认为不显示。

##### (5) PasswordChar 属性

PasswordChar 属性指定显示在文本框中的替代符，如一串 “\*” 号等，主要用于口令的输入。如果 MultiLine 属性被设为 True，则 PasswordChar 属性不起作用。

##### (6) MaxLength 属性

MaxLength 属性指定显示在文本框中的字符数，超出部分不接收，并同时发出嘟嘟声。

#### 2. 文本框控件的显示文本

文本框中显示的文本受 Text 属性控制。Text 属性可以用以下 3 种方式设置：

- 设计时在属性窗口中进行；

- 编程时通过代码设置；
- 运行时由用户输入。

通过读 Text 属性能在运行时检索文本框的当前内容。

如果要用文本框显示不希望用户更改的文本，可以把文本框的 Locked 属性设为 True，或者将文本框的 Enabled 属性设为 False。

### 3. 文本框使用示例

**【例 4-9】** 输入某学生语文、数学、英语这 3 门课程的考试成绩，计算其平均成绩。要求利用文本框实现数据的输入和输出。

设计步骤如下。

- (1) 建立用户界面，如图 4-17 所示。
- (2) 设置对象属性。各控件的属性设置参见图 4-18。

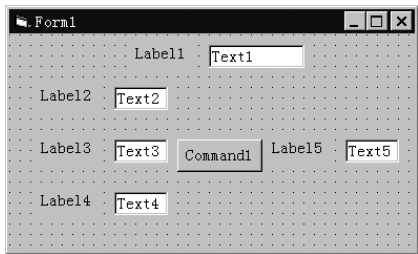


图 4-17 建立用户界面

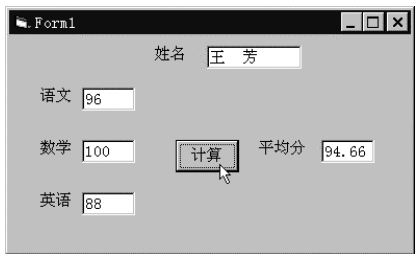


图 4-18 程序运行结果

- (3) 编写事件代码。窗体 Form 的 Load（载入）事件代码如下：

```
Private Sub Form_Load()  
    Text1.Text = "" : Text2.Text = ""           ' 设置文本框的初值，目的是清空其中的内容  
    Text3.Text = "" : Text4.Text = ""  
    Text5.Text = ""  
    Text5.Locked = True                        ' 使 Text5 不可编辑  
End Sub
```

编写“计算”命令按钮 Command1 的 Click 事件代码。由于 Text 控件默认为字符串，所以要把用到的数值型变量声明为 Single（单精度）型数据。代码如下：

```
Private Sub Command1_Click()  
    Dim a As Single, b As Single, c As Single  
    a = Val(Text2.Text)                        ' Val 函数将字符型数据转换为数值型数据  
    b = Val(Text3.Text)  
    c = Val(Text4.Text)  
    Text5.Text = (a + b + c)/3                 ' 求 3 个数的平均值  
End Sub
```

运行程序，结果如图 4-18 所示。

### 4. 多行文本框

在默认情况下，文本框只能显示单行文本，而且不显示滚动条。如果文本长度超出可用

空间，则只能显示部分文本。

如果需要使文本框能够显示多行文本，可以修改文本框的 MultiLine 和 ScrollBars 属性，但是这两种属性只能在属性窗口中修改。

(1) MultiLine 属性

当 MultiLine 属性为 True 时，文本框中可以输入或显示多行文本，同时具有文字处理器的自动换行功能，即输入的文本超出文本框宽度时，会自动换行。按 Ctrl+Enter 组合键可插入一空行。

(2) ScrollBars 属性

当 MultiLine 属性为 True 时，ScrollBars 属性才有效。ScrollBars 属性值如下。

- 0 - None：无滚动条。
- 1 - Horizontal：加水平滚动条。
- 2 - Vertical：加垂直滚动条。
- 3 - Both：同时加水平和垂直滚动条。

ScrollBars 属性的默认值为 0-None。

如果没有水平方向的滚动条，文本框中的文本会自动按字换行。当加入了水平滚动条以后，文本框内的自动换行功能会自动消失，只有按 Enter 键才能换行。

【例 4-10】 修改文本框的 MultiLine 属性和 ScrollBars 属性。

建立 4 个文本框，它们的有关属性见表 4-4。

表 4-4 属性设置

对 象	属 性	属 性 值	说 明
Text1	MultiLine	False	在默认情况下，单行文本，无滚动条
	ScrollBar	0-None	
Text2	MultiLine	True	显示多行文本
	ScrollBar	0-None	无滚动条
Text3	MultiLine	True	显示多行文本
	ScrollBar	1-Horizontal	有水平滚动条
Text4	MultiLine	True	使文本框在运行时显示多行文本
	ScrollBar	2-Vertical	有垂直滚动条

修改属性值后的结果，如图 4-19 所示。

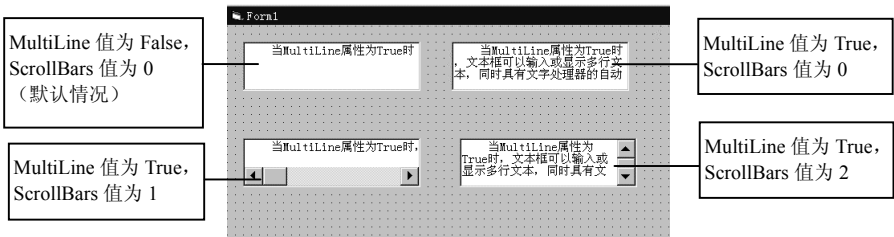


图 4-19 MultiLine 属性和 ScrollBars 属性示例

## 4.4.2 焦点与 Tab 键序

### 1. 焦点

在了解 VB 控件之前，首先要了解与控件紧密相关的焦点概念。焦点是控件接收用户鼠标或键盘动作的能力。当对象具有焦点时，可接收用户的输入。在 Windows 环境中，任一时刻都可以同时运行多个程序，但只有具有焦点的应用程序才有活动标题栏（蓝色），也只有具有焦点的程序才能接收用户输入（键盘或鼠标的动作）。

例如，当文本框具有焦点时，用户输入的数据才会出现在文本框中。

### 2. 利用 SetFocus 方法设置焦点

当控件的 Visible 和 Enabled 属性值为 True 时，控件才能接收焦点。Visible 属性决定了对象在屏幕上是否可见；Enabled 属性允许对象响应由用户产生的事件，如键盘和鼠标事件。

但是，并非所有控件都具有接收焦点的能力，如：Frame, Label, Menu, Line, Image, Timer 等控件均不能接收焦点。而且只有不包含任何可接收焦点的控件的窗体，才能接收焦点。

当对象得到或失去焦点时，会产生 GotFocus 事件或 LostFocus 事件。GotFocus 事件发生在对象得到焦点时，LostFocus 事件发生在失去焦点时。使用以下方法可以将焦点赋予对象：

- 运行时选择对象；
- 运行时用快捷键选择对象；
- 在代码中使用 SetFocus 方法，其格式为：

**〈对象〉.SetFocus**

大多数的控件得到或失去焦点时的外观是不相同的，例如，命令按钮得到焦点后周围会出现一个虚线框，文本框得到焦点后会出现闪烁的光标等。

**【例 4-11】** 修改例 4-9，编写窗体的 Activate（控件激活）事件代码，设置焦点。

在代码中调用 SetFocus 方法，使得程序开始时光标（焦点）位于输入框 Text1 中，代码如下：

```
Private Sub Form_Activate()
```

```
Text1.SetFocus
```

```
End Sub
```

另外，在“计算”按钮的 Click 事件代码中调用 SetFocus 方法，可以使光标重新回到输入框 Text1，代码如下：

```
Private Sub Command1_Click()
```

```
Dim a As Single, b As Single, c As Single
```

```
a = Val(Text2.Text) ' Val 函数将字符型数据转换为数值型数据
```

```
b = Val(Text3.Text)
```

```
c = Val(Text4.Text)
```

```
Text5.Text = (a + b + c)/3 ' 求 3 个数的平均值
```

```
Text1.SetFocus ' 设置焦点
```

```
End Sub
```

### 3. 程序运行时改变焦点的方法

程序运行时，用户可以通过下列方法之一改变焦点：



- 用鼠标单击对象；
- 按 Tab 键或 Shift+Tab 键在当前窗体上的各对象之间巡回移动焦点；
- 按热键选择对象。

#### 4. Tab 键序

所谓 Tab 键序，是指当用户按下 Tab 键时，焦点在控件间移动的顺序。每个窗体都有自己的 Tab 键序。在默认状态下，Tab 键序与建立这些控件的顺序相同。例如，在窗体上先后建立 3 个命令按钮 C1、C2 和 C3，程序启动时 C1 首先获得焦点。当用户按下 Tab 键时，焦点依次向 C2、C3 转移，如此这般往复循环。控制 Tab 键序的属性有如下两个。

##### (1) TabIndex 属性

TabIndex 属性决定控件接收焦点的顺序。当在窗体上画出第一个控件时，VB 分配给控件的 TabIndex 属性默认值为 0，第二个为 1，第三个为 2，……，其余类推。

用户在程序运行中按 Tab 键时，焦点将根据 TabIndex 属性值所指定的焦点移动顺序移到下一控件。

如果希望更改 Tab 键序，例如，希望焦点直接从 C1 转移到 C3，可以通过设置 TabIndex 属性来改变一个控件的 Tab 键序。

注意：不能获得焦点的控件及无效的和不可见的控件，不具有 TabIndex 属性，因而不包含在 Tab 键序中，按 Tab 键时这些控件将被跳过。

##### (2) TabStop 属性

TabStop 属性决定焦点是否能够停在该控件上。通常，运行时按 Tab 键能选择键序中的每一个控件。将控件的 TabStop 属性设为 False，便可将此控件从键序中删除，但仍然保持它在实际 Tab 键序中的位置，只不过在按 Tab 键时这个控件将被跳过。

### 4.4.3 框架控件

框架（Frame）控件是一种容器控件。在框架控件内的控件可以随框架一起移动，并且受框架控件某些属性（Visible、Enabled 等）的控制。

#### 1. 使用框架控件分组

在设计界面时，经常使用框架控件对其他控件进行分组，以使界面更清晰明了。在一般情况下，不需要响应框架控件的事件。

在框架控件中，需要修改的属性一般为 Name、Caption 或 Font 属性。

使用框架控件将其他控件分组的方法有两个：

- 先画出框架控件并激活它，再加入其中的控件，这样可使框架控件及其上的控件一起移动；
- 如果要用框架控件将现有的控件分组，可先选定所有控件，将它们剪切到剪贴板上，然后选定框架控件并将剪贴板上的控件粘贴到框架控件上。

#### 2. 框架控件使用示例

**【例 4-12】** 如图 4-20 所示，在 3 个文本框中分别输入时、分、秒，转化成秒数后输出。

**【分析】** 设通过文本框控件输入的时数为  $h$ ，分数为  $m$ ，秒数为  $s$ ，则利用公式

$$x = h \times 3600 + m \times 60 + s$$

可以计算出合计秒数  $x$ 。

设计步骤如下。

（1）建立用户界面。新建一个工程，进入窗体设计器，在窗体中增加一个框架控件 Frame1，一个命令按钮 Command1 和一个标签 Label1。在 Frame1 上添加 3 个文本框控件 Text1~Text3，如图 4-21 所示。

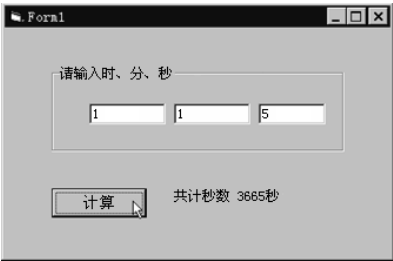


图 4-20 计算秒数

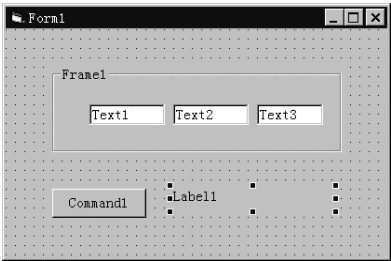


图 4-21 用户界面

（2）设置对象属性，见表 4-5。设置属性后的窗体如图 4-20 所示。

表 4-5 属性设置

对 象	属 性	属 性 值	说 明
Frame1	Caption	请输入时、分、秒	框架的标题
Command1	Caption	计算	按钮的标题
Text1~Text3	Text	(空)	文本框的内容
Label1	Caption	(空)	标签的标题

（3）编写事件代码。“计算”命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()  
    Dim h As Integer, m As Integer, s As Integer      ' 定义时、分、秒的数据类型为整型  
    Dim x As Long  
    h = Val(Text1.Text)                             ' 通过 Text1 输入时数  
    m = Val(Text2.Text)                             ' 通过 Text2 输入分数  
    s = Val(Text3.Text)                             ' 通过 Text3 输入秒数  
    x = h * 3600 + m * 60 + s                        ' 计算秒数  
    Label1.Caption = "共计秒数" & Str(x) & "秒"      ' 输出到 Label1  
End Sub
```

运行程序，结果如图 4-20 所示。

### 4.5 使用对话框

在图形用户界面中，对话框（DialogBox）是程序与用户交互的另一种主要途径。对话框分为两种。

- 输入框（InputBox）：可以输入信息。
- 消息框（MsgBox）：可以显示信息。

4.5.1 输入框（InputBox）函数

输入框函数用来显示一个能接收用户输入数据的对话框，并返回用户在对话框中输入的信息。

1. 输入框的语法格式及使用说明

输入框的语法格式为：

```
〈变量〉 = InputBox(〈信息内容〉[, 〈对话框标题〉][, 〈默认内容〉])
```

【说明】

- ① 〈信息内容〉为在对话框中出现的文本。在〈信息内容〉中使用硬回车符 CHR(13)可以使文本换行。对话框的高度和宽度随着〈信息内容〉的增加而增加，最多可有 1024 个字符。
- ② 〈对话框标题〉用来指定对话框的标题。
- ③ 〈默认内容〉可以指定输入框的文本框中显示的默认文本。若用户单击“确定”按钮，文本框中的文本将返回到〈变量〉中；若用户单击“取消”按钮，返回的将是一个 0 长度的字符串。
- ④ 如果省略了某些可选项，其间的逗号分隔符不可省略。

2. 输入框使用示例

【例 4-13】“鸡兔同笼”问题。鸡有 2 只脚，兔有 4 只脚，如果已知鸡和兔的总头数为  $h$ ，总脚数为  $f$ 。问笼中鸡和兔各有多少只？

【分析】设笼中有鸡  $x$  只，兔  $y$  只，由条件可得方程组：

$$\begin{cases} x + y = h \\ 2x + 4y = f \end{cases}$$

解方程组得：

$$\begin{cases} x = \frac{4h - f}{2} \\ y = \frac{f - 2h}{2} \end{cases}$$

设计步骤如下。

（1）建立用户界面并设置对象属性。新建一个工程，进入窗体设计器，首先增加 3 个标签 Label1~Label3 和一个命令按钮 Command1。其属性设置如图 4-22 所示。

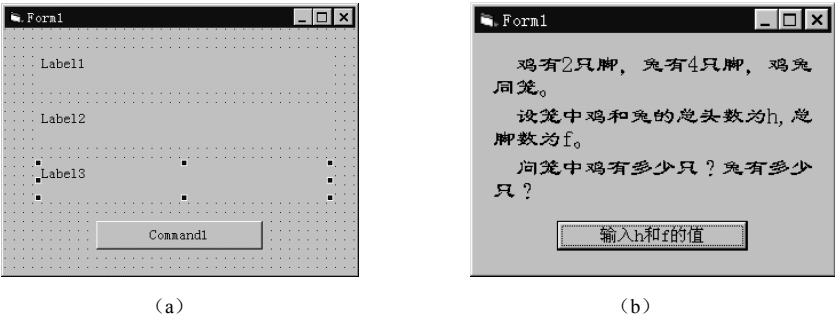


图 4-22 “鸡兔同笼”问题

(2) 编写事件代码。“输入 h 和 f 的值” 命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()  
    Dim h As Integer, f As Integer  
    h = Val(InputBox("鸡和兔的总头数", "请输入", 0))  
    f = Val(InputBox("鸡和兔的总脚数（偶数）", "请输入", 0))  
  
    x = (4 * h - f) / 2  
    y = (f - 2 * h) / 2  
  
    Label2.Caption = " 设笼中鸡和兔的总头数为" & h & ", 总脚数为" & f & "。"  
    Label3.Caption = " 则笼中鸡有" & x & "只,兔有" & y & "只。"  
  
End Sub
```

运行程序后，单击命令按钮将依次弹出两个输入框（见图 4-23）来接收输入的数据。

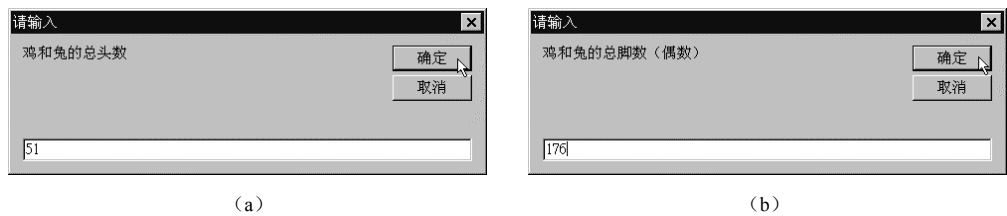


图 4-23 输入框

输出结果，如图 4-24 所示。

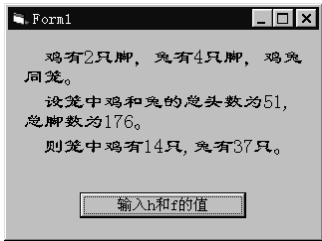


图 4-24 输出结果

### 4.5.2 消息框（MsgBox）函数

消息框函数在对话框中显示信息，等待用户单击按钮，并返回一个整数以表明用户单击了哪个按钮。

#### 1. 消息框的语法格式及使用说明

消息框的语法格式为：

```
〈变量〉 = MsgBox(〈消息内容〉 [, 〈对话框类型〉 [, 〈对话框标题〉 ]])
```

##### 【说明】

- ① 〈消息内容〉为在对话框中出现的文本。在〈消息内容〉中使用硬回车符 CHR(13)可以使文本换行。对话框的高度和宽度随着〈消息内容〉的增加而增加，最多可有 1024 个字符。
- ② 〈对话框类型〉用来指定对话框中出现的按钮和图标，一般有 3 个参数。其取值和含义见表 4-6、表 4-7 和表 4-8。这 3 种参数值可以相加以达到所需要的样式。

表 4-6 参数 1——出现按钮

值	常 量	说 明
0	vbOKOnly	确定按钮
1	vbOKCancel	确定和取消按钮
2	vbAbortRetryIgnore	终止、重试和忽略按钮
3	vbYesNoCancel	是、否和取消按钮
4	vbYesNo	是和否按钮
5	vbRetryCancel	重试和取消按钮

表 4-7 参数 2——图标类型

值	常 量	说 明
16	vbCritical	停止图标
32	vbQuestion	问号（?）图标
48	vbExclamation	感叹号（!）图标
64	vbInformation	消息图标

表 4-8 参数 3——默认按钮

值	常 量	说 明
0	vbDefaultButton1	默认按钮为第一按钮
256	vbDefaultButton2	默认按钮为第二按钮
512	vbDefaultButton3	默认按钮为第三按钮

③ 〈对话框标题〉用来指定对话框的标题。下述代码将显示如图 4-25 所示的对话框：

```
msg = MsgBox("请确认输入的数据是否正确！", 3 + 32 + 0, "数据检查")
```

④ MsgBox()函数的返回值指明了在对话框中选择哪一个按钮，见表 4-9。

表 4-9 MsgBox 函数的返回值

返 回 值	常 量	按 钮
1	vbOK	确定
2	vbCancel	取消
3	vbAbort	终止
4	vbRetry	重试
5	vbIgnore	忽略
6	vbYes	是
7	vbNo	否

⑤ 代码中的值可以是数值，也可以是常量。

⑥ 如果省略了某些可选项，其间的逗号分隔符不可省略。

⑦ 若不需要返回值，可以使用下面的 MsgBox 命令形式：

```
MsgBox 〈信息内容〉 [, 〈对话框类型〉] [, 〈对话框标题〉] ]
```

2. 消息框使用示例

在程序运行的过程中，有时需要显示一些简单的信息，如警告或错误提示等，此时可以利用消息对话框来显示这些内容。当用户接收到信息后，可以单击按钮来关闭对话框，并返

回单击的按钮值。

【例 4-14】 修改例 4-12，利用消息框进行数据输出。  
设计步骤如下。

- (1) 修改用户界面。删除标签 Label1。
- (2) 修改事件代码。

```
Private Sub Command1_Click()  
    Dim h As Integer, m As Integer, s As Integer  
    Dim x As Long  
    h = Val(Text1.Text)  
    m = Val(Text2.Text)  
    s = Val(Text3.Text)  
    x = h * 3600 + m * 60 + s  
    MsgBox "共计秒数" & Str(x) & "秒", 1 + 64 + 0, "秒数换算" ' 利用消息框输出  
End Sub
```

运行程序，结果如图 4-26 所示。



图 4-25 消息对话框

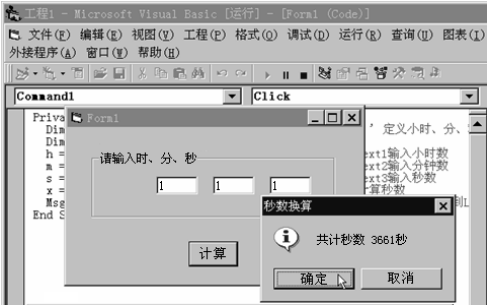


图 4-26 利用消息框输出结果

## 习题 4

### 一、选择题

- 4.1 输入代码时，VB 可以自动检测 ( ) 错误。  
A) 语法                      B) 编译                      C) 运行                      D) 逻辑
- 4.2 在 VB 中，要将一个窗体加载到内存进行预处理但不显示，应使用的语句是 ( )。  
A) Load                      B) Show                      C) Hide                      D) Unload
- 4.3 以下能在窗体 Form1 的标题栏中显示"Visual Basic 窗体"的语句是 ( )。  
A) Form1.Name="Visual Basic 窗体"                      B) Form1.Title="Visual Basic 窗体"  
C) Form1.Caption="Visual Basic 窗体"                      D) Form1.Text="Visual Basic 窗体"
- 4.4 对下列程序段，说法正确的是 ( )  
Text1.Top=2000 : Text1.Left=800  
A) Text 对象的左边距窗体的左边的距离为 800twip，上边距窗体的上边的距离为 2000twip  
B) Text1 对象的左边距屏幕的左边的距离为 800twip，上边距屏幕的上边的距离为 2000twip

C) Text1 对象的宽度为 2000twip, 高度为 800twip

D) Text1 对象的高度为 800 点, 宽度为 2000 点

4.5 在运行程序时, 在文本框中输入新的内容, 或者在程序代码中改变 Text 的属性值, 相应会触发 ( ) 事件。

A) GotFocus                      B) Click                      C) Change                      D) DblClick

4.6 单击窗体上的关闭按钮时, 触发的事件是 ( )。

A) Form\_Initialize()    B) Form\_Load()              C) Form\_Unload()              D) Form\_Click()

4.7 可以实现从键盘输入一个作为双精度变量 a 的值的语句是 ( )。

A) a=InputBox()                      B) a=InputBox("请输入一个值")

C) a=Val(InputBox("请输入一个值"))              D) a=Val(InputBox())

4.8 在属性窗口中设置 ( ) 属性, 可以把指定的图形放入当前对象中。

A) CurrentY                      B) Picture                      C) CurrentX                      D) Stretch

4.9 用于将屏幕上的对象分组的控件是 ( )。

A) 列表框                      B) 组合框                      C) 标签                      D) 框架

4.10 要将名为 MyForm 的窗体显示出来, 正确的使用方法是 ( )。

A) MyForm.Show              B) Show.MyForm              C) MyForm Load              D) MyForm Show

4.11 如果要窗体中的某个命令按钮设置成无效状态, 应设置命令按钮的 ( ) 属性。

A) Value                      B) Visible                      C) Enabled                      D) Default

4.12 能够获得一个文本框中被选取文本的内容的属性是 ( )。

A) Text                      B) Length                      C) SelText                      D) SelStart

4.13 用 InputBox 函数设计的对话框, 其功能是 ( )。

A) 只能接收用户输入的数据, 但不会返回任何信息

B) 能接收用户输入的数据, 并能返回用户输入的信息

C) 既能用于接收用户输入的信息, 又能用于输出信息

D) 专门用于输出信息

## 二、填空题

4.14 当对象得到焦点时, 会触发\_\_\_\_事件; 当对象失去焦点时, 会触发\_\_\_\_事件。

4.15 下列语句的输出结果为\_\_\_\_\_。

```
Print Format $(5689.36,"000,000.000")
```

4.16 新建一个工程, 内有两个窗体, 窗体 Form1 上有一个命令按钮 Command1, 单击该按钮, Form1 窗体消失, 显示 Form2 窗体, 试补全程序。

```
Private Sub Command1_Click()
```

```
_____  
Form2.____
```

```
End Sub
```

4.17 设有如下程序段:

```
a$="BeijingShanghai" : b$=Mid(a$,InStr(a$,"g")+1)
```

执行上面的程序段后，变量 b\$ 的值为\_\_\_\_\_。

4.18 为了使一个窗体从屏幕上消失但仍在内存中，所使用的方法或语句为\_\_\_\_\_。

4.19 在文本框中要使输入的所有字符显示为\*号，应设置\_\_\_\_\_属性为"\*"。

### 三、编程题

4.20 设计工程，已知圆的半径  $r$ ，求圆面积  $S$ 。

4.21 已知平面坐标系中两点的坐标，求两点间的距离。

4.22 设计工程，输出在指定范围内的 3 个随机数，范围在文本框中输入。

4.23 设某职工应发工资  $x$  元，试求各种面额钞票总张数最少的付款方案。

4.24 使用大小写转换函数设计程序，实现在文本框中输入英文字母，按“转大写”按钮，文本变为大写；按“转小写”按钮，文本变为小写。

4.25 在文本框中输入 3 种商品的单价、购买数量，计算并输出所用的总金额。

4.26 在文本框中输入弧度值，将弧度换算为角度值（度、分、秒）的形式，然后输出。例如，弧度值为 1.474919573，化为角度的方法为：

（1）先将弧度值变成十进制数，即  $1.474\ 919\ 573 \times (180/\pi) = 84.506\ 666\ 65$ 。

（2）去掉整数部分 84，余 0.506 666 65。

（3） $0.506\ 666\ 65 \times 60 = 30.399\ 999$ 。

（4）去掉 30，余 0.399 999。

（5） $0.399\ 999 \times 60 = 23.999\ 94 \approx 24$ 。

（6）最后将 84, 30, 24 拼 接成  $84^\circ 30' 24''$ 。



## 第 5 章 选择结构程序设计

在日常生活和工作中，经常需要根据不同的情况，采用不同的工作方法。同样，在程序设计过程中，也常常需要根据给定的条件，采取不同的分支操作。

选择结构的特点是：根据所给定的条件为真（即条件成立）与否，决定从各实际可能的不同分支中执行某一支的相应操作，并且在任何情况下总有“无论分支多寡，必择其一”的特性。VB 提供了多种形式的条件语句来实现选择结构，即对条件进行判断，并根据判断结果，选择执行不同的分支。

### 5.1 If 语句

单条件选择结构是最常用的双分支选择结构。

#### 5.1.1 单行结构条件语句 If...Then...Else

##### 1. 单行条件语句 If 的语法格式

单行条件语句比较简单，其语法格式为：

**If 〈条件〉 Then [ 〈语句组 1〉 ] | Else 〈语句组 2〉 ]**

##### 【说明】

① 〈条件〉可以是关系表达式、逻辑表达式或数值表达式。如果以数值表达式作为条件，则非 0 值为真，0 为假。

② 单行条件语句的执行过程为：判断 〈条件〉，若为真，则执行 〈语句组 1〉；否则执行 Else 后面的 〈语句组 2〉。

③ 如果没有 Else 子句，〈语句组 1〉为必要参数，在 〈条件〉为真时执行；〈条件〉为 False 时，什么都不做，执行 If 语句下面的语句。

##### 2. 单行 If 语句的使用示例

**【例 5-1】** 编写程序，任意输入一个整数，判定该整数的奇偶性。

**【分析】** 判断某整数的奇偶性，就是检查该数是否能被 2 整除。若能被 2 整除，该数为偶数；否则为奇数。被 2 整除，可以利用 Mod 运算来完成。

设计步骤如下。

(1) 建立用户界面，如图 5-1 所示。

(2) 设置对象属性，见表 5-1。

(3) 编写事件代码。

“判定”命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()
```

```
Dim x As Integer
x = Val(Text1.Text)
Label2.FontSize = 20
If x Mod 2 = 0 Then Label2.Caption = "偶数" Else Label2.Caption = "奇数"
End Sub
```

表 5-1 属性设置

对 象	属 性	属 性 值	说 明
Form	Caption	判定整数的奇偶性	
	BorderStyle	1-Fixed Dialog	固定对话框，包含控制菜单框和标题栏，不包含最大化按钮和最小化按钮，不能改变尺寸
Label1	Caption	请输入一个整数:	
Label2	Caption		空
Command1	Caption	判定	
Command2	Caption	清除	
Command3	Caption	结束	
Text1	Text	""	空

“清除”命令按钮 Command2 的 Click 事件代码如下：

```
Private Sub Command2_Click()
    Text1.Text = ""
End Sub
```

“结束”命令按钮 Command3 的 Click 事件代码如下：

```
Private Sub Command3_Click()
    Unload Me
End Sub
```

运行程序，结果如图 5-2 所示。

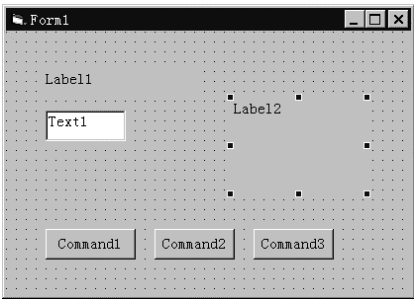


图 5-1 用户界面



图 5-2 运行程序

【例 5-2】  $x, y$  关系如下：

$$y = \begin{cases} 1 + x & (x \geq 0) \\ 1 - 2x & (x < 0) \end{cases}$$

设计程序，输入  $x$ ，可计算出  $y$  的值，程序界面如图 5-3 所示。

【分析】该题是数学中的一个分段函数，它表示：当  $x \geq 0$  时，用公式  $y = 1 + x$  来计算  $y$

的值；当  $x < 0$  时，用公式  $y = 1 - 2x$  来计算  $y$  的值。在选择条件时，既可以选择  $x \geq 0$  作为条件，也可以选择  $x < 0$  作为条件。在这里，选  $x \geq 0$  作为选择条件。这时，当  $x \geq 0$  为真时，执行  $y = 1 + x$ ；为假时，执行  $y = 1 - 2x$ 。

设计步骤如下。

- (1) 建立应用程序用户界面并设置对象属性，参见图 5-3。
- (2) 编写程序代码。

“计算”命令按钮 Command1 的 Click 事件代码为：

```
Private Sub Command1_Click()  
    Dim x As Single, y As Single  
    x = Val(Text1.Text)  
    If x >= 0 Then y = 1 + x Else y = 1 - 2 * x  
    Text2.Text = y  
End Sub
```

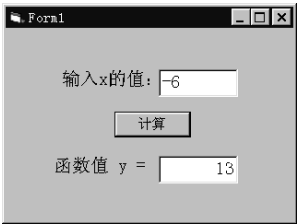


图 5-3 计算函数的值

5.1.2 块结构条件语句 If...Then...Else...End If

使用单行 If 语句，可以满足许多选择结构程序设计的需要，但是当 Then 部分和 Else 部分包含较多内容时，在一行中就难以容纳所有命令了，即使能够容纳，也将使程序结构变得不清晰。为此，VB 提供了块 If 语句，将一个选择结构用多个语句行来实现。

1. 块 If 语句的语法格式

块 If 语句又称为多行 If 语句，其语法格式为：

```
If <条件> Then  
    <语句组 1>  
[Else  
    <语句组 2> ]  
End If
```

【说明】

- ① 在块结构中，If 语句必须是第一行语句。If 块必须以一个 End If 语句结束。
- ② 当程序运行到 If 块时，首先测试 <条件>，如果为 True，则执行 Then 之后的 <语句组 1>；如果为 False，且有 Else 子句，则执行 Else 部分的 <语句组 2>。执行完 Then 或 Else 之后的语句组后，从 End If 之后的语句继续执行。
- ③ Else 子句是可选的。

2. 块 If 语句使用示例

【例 5-3】 将例 5-1 中命令按钮 Command1 的 Click 事件代码改写为块 If 语句。代码如下：

```
Private Sub Command1_Click()  
    Dim x As Integer  
    x = Val(Text1.Text)  
    Label2.FontSize = 20
```

```

If x Mod 2 = 0 Then
    Label2.Caption = "偶数"
Else
    Label2.Caption = "奇数"
End If

```

**End Sub**

【例 5-4】 将例 5-2 中命令按钮 Command1 的 Click 事件代码改为多行 If 语句。代码如下：

```

Private Sub Command1_Click()
    Dim x As Single, y As Single
    x = Val(Text1.Text)
    If x >= 0 Then
        y = 1 + x
    Else
        y = 1 - 2 * x
    End If
    Text2.Text = y
End Sub

```

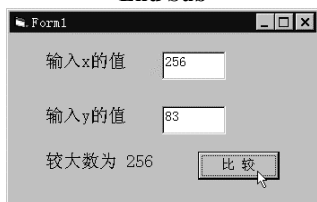


图 5-4 比较两数大小

【例 5-5】 已知两个数  $x$  和  $y$ ，设计程序，比较它们的大小，并输出较大数。

设计步骤如下。

(1) 建立用户界面并设置对象属性，如图 5-4 所示。

(2) 编写事件代码。“比较”命令按钮 Command1 的 Click 事件代码为：

```

Private Sub Command1_Click()
    Dim x As Single, y As Single
    x = Val(Text1.Text)
    y = Val(Text2.Text)
    If x < y Then
        t = x
        x = y
        y = t
    End If
    Label3.Caption = "较大数为" & Str(x)
End Sub

```

**End Sub**

运行程序，结果如图 5-4 所示。

### 5.1.3 使用 IIf 函数

使用 IIf 函数也可以实现一些比较简单的选择结构。IIf 函数的语法结构为：

**IIf(〈条件表达式〉,〈真部分〉,〈假部分〉)**

**【说明】**

- ① 〈条件表达式〉可以是关系表达式、布尔表达式或数值表达式。如果用数值表达式作为条件，则非 0 为真，0 为假。
- ② 〈真部分〉是当条件表达式为真时函数返回的值，可以是任何表达式。
- ③ 〈假部分〉是当条件表达式为假时函数返回的值，可以是任何表达式。
- ④ 语句

**y = IIf(〈条件表达式〉,〈真部分〉,〈假部分〉)**

相当于

**If 〈条件表达式〉 then y = 〈真部分〉 Else y = 〈假部分〉**

**【例 5-6】** 使用 IIf 函数，例 5-2 中命令按钮 Command1 的 Click 事件代码可以改写为：

```
Private Sub Command1_Click()  
    Dim x As Single, y As Single  
    x = Val(Text1.Text)  
    y = IIf(x >= 0, 1 + x, 1 - 2 * x)  
    Text2.Text = y  
End Sub
```

**5.1.4 If 语句的嵌套**

**1. If 语句的嵌套**

If 语句的嵌套是指 If 或 Else 后面的语句块中又包含 If 语句。语法格式如下：

```
If 〈条件 1〉 Then  
    If 〈条件 2〉 Then  
        .....  
    End If  
    .....  
End If
```

**【例 5-7】** 铁路托运行李，从甲地到乙地，规定每张客票托运费计算方法是：行李重量不超过 50 千克时，每千克 0.25 元；超过 50 千克而不超过 100 千克时，其超过部分每千克 0.35 元；超过 100 千克时，其超过部分每千克 0.45 元。编写程序，输入行李重量，计算并输出托运的费用（如图 5-5 所示）。

**【分析】** 设行李重量为  $w$  千克，应付托运费为  $x$  元，则托运费公式为：

$$x = \begin{cases} 0.25w & (w \leq 50) \\ 0.25 \times 50 + 0.35 \times (w - 50) & (50 < w \leq 100) \\ 0.25 \times 50 + 0.35 \times 50 + 0.45 \times (w - 100) & (w > 100) \end{cases}$$

设计步骤如下。

- (1) 建立应用程序用户界面并设置对象属性，如图 5-5 所示。

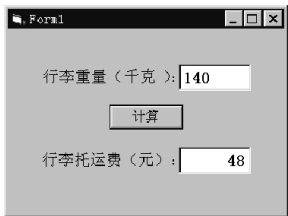


图 5-5 计算托运费

(2) 编写程序代码。写出命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()  
    Dim w As Single, x As Single  
    w = Val(Text1.Text)  
    If w <= 50 Then  
        x = 0.25 * w  
    Else  
        If w <= 100 Then  
            x = 0.25 * 50 + 0.35 * (w-50)  
        Else  
            x = 0.25 * 50 + 0.35 * 50 + 0.45 * (w-100)  
        End If  
    End If  
    Text2.Text = x  
End Sub
```

【例 5-8】 某百货公司为了促销，采用购物打折的优惠办法，如图 5-6 所示。每位顾客一次购物：

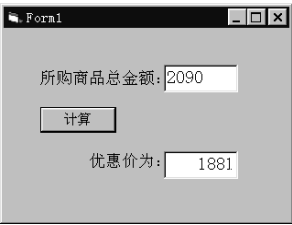


图 5-6 计算优惠价

- (1) 1000 元以上者，按九五折优惠。
- (2) 2000 元以上者，按九〇折优惠。
- (3) 3000 元以上者，按八五折优惠。
- (4) 5000 元以上者，按八〇折优惠。

编写程序，输入购物款数，计算并输出优惠价。

【分析】设购物款数为  $x$  元，优惠价为  $y$  元，则优惠付款公式为：

$$y = \begin{cases} x & (x < 1000) \\ 0.95x & (1000 \leq x < 2000) \\ 0.9x & (2000 \leq x < 3000) \\ 0.85x & (3000 \leq x < 5000) \\ 0.8x & (x \geq 5000) \end{cases}$$

设计步骤如下。

- (1) 建立应用程序用户界面并设置对象属性，如图 5-6 所示。
- (2) 编写程序代码。命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()  
    Dim x As Single, y As Single  
    x = Val(Text1.Text)  
    If x < 1000 Then  
        y = x ' 1000 元以下不优惠  
    Else  
        If x < 2000 Then
```

```

        y = 0.95 * x                                ' 1000~2000 元，九五折
Else
    If x < 3000 Then
        y = 0.9 * x                                ' 2000~3 000 元，九〇折
    Else
        If x < 5000 Then
            y = 0.85 * x                            ' 3000~5000 元，八五折
        Else
            y = 0.8 * x                            ' 5000 元以上，八〇折
        End If
    End If
End If
Text2.Text = y
End Sub

```

## 2. If 语句的嵌套格式 ElseIf

从例 5-8 可以看出，程序代码中出现了多层 If 语句嵌套，这样影响了程序的清晰性。在这种情况下，可以使用 If 语句的嵌套格式 ElseIf 进行设计，即 ElseIf 的块 If 语句，使程序简化易懂。If 语句嵌套格式的语法格式为：

```

If 〈条件 1〉 Then
    〈语句组 1〉
ElseIf 〈条件 2〉 Then
    〈语句组 2〉
    .....
[Else
    〈语句组  $n+1$ 〉 ]
End If

```

### 【说明】

① 在 If 块中，Else 子句和 ElseIf 子句都是可选的，可以放置任意多个 ElseIf 子句，但是都必须在 Else 子句之前。

② 执行过程：当程序运行到 If 块时，测试〈条件 1〉，如果为真，则执行 Then 之后的语句；如果为假，则依次计算每个 ElseIf 部分的条件式（如果有的话）并加以测试。如果找到某个为真的条件，则紧接在其相关的 Then 之后的语句组被执行。如果没有一个 ElseIf 条件为真（或者根本就没有 ElseIf 子句），则程序会执行 Else 部分的〈语句组  $n+1$ 〉。在执行完 Then 或 Else 之后的语句列后，会从 End If 之后的语句继续执行。

**【例 5-9】** 修改例 5-8 程序代码，使用带 ElseIf 的块 If 语句计算优惠价。

只需将“计算”命令按钮 Command1 的 Click 事件代码改写为：

```

Private Sub Command1_Click()

```

```

Dim x As Single, y As Single
x = Val(Text1.Text)
If x < 1000 Then
    y = x                                ' 1000 元以下不优惠
ElseIf x < 2000 Then
    y = 0.95 * x                        ' 1000~2000 元，九五折
ElseIf x < 3000 Then
    y = 0.9 * x                        ' 2000~3000 元，九〇折
ElseIf x < 5000 Then
    y = 0.85 * x                       ' 3000~5000 元，八五折
Else
    y = 0.8 * x                        ' 5000 元以上，八〇折
End If
Text2.Text = y
End Sub

```

## 5.2 多分支条件选择语句 Select Case

多分支选择结构的特点是：从多个选择结构中，选择第一个条件为真的路线作为执行的路线。

### 1. Select Case 语句的语法格式

使用 If 语句的嵌套可以实现多分支选择，但仍然不够便捷。为此，VB 提供了多分支选择语句 Select Case 来实现多分支选择。这样，当需要根据某一表达式的结果执行多种可能的动作时，使用 Select Case 语句更为简捷。

使用 Select Case 语句进行多分支选择的特点是：从多个选择分支中，选择第一个条件为真的路线作为执行路线。

Select Case 语句的语法格式为：

```

Select Case 〈测试条件〉
    [Case 〈表达式表 1〉
        〈语句组 1〉 ]
    [Case 〈表达式表 2〉
        〈语句组 2〉 ]
    .....
    [Case Else
        〈语句组 n+1〉 ]
End Select

```

#### 【说明】

- ① 〈测试条件〉为必要参数，可以是任何数值表达式或字符串表达式。



②〈语句组〉为可选参数，是一条或多条语句，当〈表达式表〉中有值且与〈测试条件〉相匹配时执行。

③ Case Else 子句用于指明其他语句组，当〈测试条件〉与所有 Case 子句的〈表达式表〉中的值都不匹配时，会执行这些语句。虽然不是必要的，但是在 Select Case 块中，最好还是加上 Case Else 语句来处理不可预见的测试条件值。如果没有 Case 值匹配测试条件，而且也没有 Case Else 语句，则程序会从 End Select 之后的语句继续执行。

### 2. Select Case 语句中〈表达式表〉的使用说明

在 Select Case 语句的 Case 子句中，〈表达式表〉为必要参数，用来测试其中是否有值与〈测试条件〉相匹配。

Case 子句中的〈表达式表〉是一个或多个表达式的列表，形式有以下 3 种。

(1) 形式一

〈表达式〉

【说明】表达式为数值表达式或字符串表达式，例如：

Case 3 \* x

(2) 形式二

〈表达式〉 To 〈表达式〉

【说明】用来指定一个值范围，较小的值要放在 To 之前，例如：

Case 1 To 10

Case "a" To "d"

(3) 形式三

Is 〈关系运算表达式〉

【说明】可以配合比较运算符来指定一个数值范围。如果没有提供，则 Is 关键字会被自动插入。例如：

Case Is < 100

当使用多个表达式的列表时，表达式与表达式之间要用逗号“,”隔开。例如：

Case 2 , 4 , 6

Case 10 To 20 , 100 To 200

Case Is < 10 , Is > 30

在 Case 子句中使用多个表达式时，所列表达式的形式可以不相同，既可以使用值，也可以使用条件或范围，还可以混合使用。例如：

Case 2 , 4 , 6 , 10 To 20 , Is > 30

### 3. Select Case 语句使用示例

【例 5-10】 用 Select Case 语句修改例 5-9 中的代码。

将“计算”命令按钮 Command1 的 Click 事件代码改写为：

```
Private Sub Command1_Click()  
    Dim x As Single, y As Single  
    x = Val(Text1.Text)  
    Select Case x
```

```

Case Is < 1000
    y = x
Case Is < 2000
    y = 0.95 * x
Case Is < 3000
    y = 0.9 * x
Case Is < 5000
    y = 0.85 * x
Case Else
    y = 0.8 * x
End Select
Text2.Text = y

```

**End Sub**

【例 5-11】 为某航空公司计算票价的優惠率。假设优惠规定如下。

(1) 在旅游的旺季 7~9 月份，如果订票数超过 20 张，票价优惠 15%；20 张以下，优惠 5%。

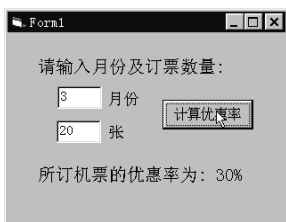


图 5-7 计算优惠价

(2) 在旅游的淡季 1~5 月份、10 月份和 11 月份，如果订票数超过 20 张，票价优惠 30%；20 张以下，优惠 20%。

(3) 其他情况一律优惠 10%。

试设计程序，根据月份和订票张数决定票价的優惠率。设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性，如图 5-7 所示。

(2) 编写程序代码。编写“计算优惠率”命令按钮 Command1

的 Click 事件代码如下：

**Private Sub Command1\_Click()**

```

Dim m As Integer, n As Integer, r As Integer
m = Val(Text1.Text)
n = Val(Text2.Text)
Select Case m
    Case Is <= 5, 10, 11
        ' 1~5 月份、10 月份和 11 月份
        If n < 20 Then r = 20 Else r = 30
        ' 超过 20 张优惠 30%，20 张以下优惠 20%
    Case 7 To 9
        ' 7~9 月份
        If n < 20 Then r = 5 Else r = 15
        ' 超过 20 张优惠 15%，20 张以下优惠 5%
    Case Else
        ' 其他情况优惠 10%
        r = 10
End Select
Label4.Caption = "所订机票的优惠率为:" & Str(r) & "%"

```

**End Sub**

运行程序，结果如图 5-7 所示。

在本例中，为了使用更加方便，还可以再增加如下代码。

窗体 Form 的 Load（载入）事件代码：

```
Private Sub Form_Load()  
    Text1.Text = Month(Date)           ' Text1 中的默认值为当前月份  
End Sub
```

文本框 Text1 的 GotFocus（获得焦点）事件代码：

```
Private Sub Text1_GotFocus()  
    Text1.SelStart = 0                 ' Text1 文本的起点位置为 0  
    Text1.SelLength = Len(Text1.Text) ' 选择文本的长度  
End Sub
```

文本框 Text1 的 KeyPress（按键）事件代码：

```
Private Sub Text1_KeyPress(KeyAscii As Integer)  
    If KeyAscii = 13 Then              ' 在 Text1 中按 Enter 键，光标跳到 Text2 中  
        If Text1.Text > 0 And Text1.Text < 13 Then  
            Text2.SetFocus              ' 设置焦点  
        End If  
    End If  
End Sub
```

文本框 Text2 的 GotFocus（获得焦点）事件代码：

```
Private Sub Text2_GotFocus()  
    Text2.SelStart = 0  
    Text2.SelLength = Len(Text2.Text)  
End Sub
```

文本框 Text2 的 KeyPress（按键）事件代码：

```
Private Sub Text2_KeyPress(KeyAscii As Integer)  
    If KeyAscii = 13 Then              ' 在 Text2 中按 Enter 键，光标跳到 Command1 中  
        If Text2.Text > 0 Then Command1.SetFocus  
    End If  
End Sub
```

**【例 5-12】** 任给定一年，判断该年是否为闰年，并根据给出的月份来判断是什么季节和该月有多少天。闰年的条件是：年份能被 4 整除但不能被 100 整除，或者能被 400 整除。

**【分析】** 根据闰年条件可得出闰年的逻辑表达式

```
(y Mod 4=0 And y Mod 100<>0) Or (y Mod 400=0)
```

每月的天数可根据月份来判定。

设计步骤如下。

- （1）建立用户界面，参见图 5-8。
- （2）设置对象属性，见表 5-2。

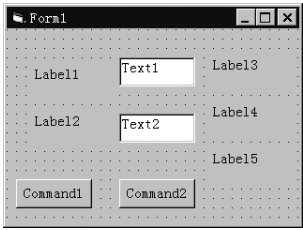


图 5-8 用户界面

表 5-2 属性设置

对 象	属 性	属 性 值	说 明
Form1	Caption	判断闰年、季节、天数	
Command1	Caption	开始	
	Default	True	默认命令按钮
Command2	Caption	清除	
Text1	Text		清空
Text2	Text		清空
Label1	Caption	年份:	
Label2	Caption	月份:	
Label3	Caption		空
	BorderStyle	1-Fixed Single	边框样式
Label4	Caption		空
	BorderStyle	1-Fixed Single	边框样式
Label5	Caption		空
	BorderStyle	1-Fixed Single	边框样式

(3) 编写事件代码。

编写“开始”命令按钮 Command1 的 Click 事件代码:

```
Private Sub Command1_Click()  
    Dim y As Integer, m As Integer, days As Integer  
    Dim leapyear As Boolean                ' 闰年标记  
    y = Val(Text1.Text)                   ' y 为年份  
    m = Val(Text2.Text)                   ' m 为月份  
    ' 判断闰年  
    If (y Mod 4 = 0 And y Mod 100 <> 0) Or (y Mod 400 = 0) Then  
        leapyear = True                   ' True 为闰年  
        Label3.Caption = "闰年"  
    Else  
        leapyear = False                  ' False 为非闰年  
        Label3.Caption = "不是闰年"  
    End If  
    ' 判断季节  
    Select Case m  
        Case 3 To 5  
            Label4.Caption = "春季(Spring)"  
        Case 6 To 8  
            Label4.Caption = "夏季(Summer)"  
        Case 9 To 11  
            Label4.Caption = "秋季(Autumn)"  
        Case 12, 1, 2
```

```
Label4.Caption = "冬季(Winter)"
End Select
' 判断月中天数
Select Case m
Case 1, 3, 5, 7, 8, 10, 12
Label5.Caption = "31 天!"           ' 大月为 31 天
Case 4, 6, 9, 11
Label5.Caption = "30 天!"           ' 小月为 30 天
Case 2                               ' 判断 2 月份的天数
If leapyear Then
days = 29                           ' 闰年为 29 天
Else
days = 28                           ' 非闰年为 28 天
End If
Label5.Caption = Str(days) + "天"
End Select
Text1.SetFocus
```

**End Sub**

编写“清除”命令按钮 Command2 的 Click 事件代码：

```
Private Sub Command2_Click()
Text1.Text = ""
Text2.Text = ""
Label3.Caption = ""
Label4.Caption = ""
Label5.Caption = ""
End Sub
```

运行程序，输入年份和月份后，单击“开始”按钮，将输出结果，如图 5-9 所示。单击“清除”按钮，清除文本框中输入的内容。

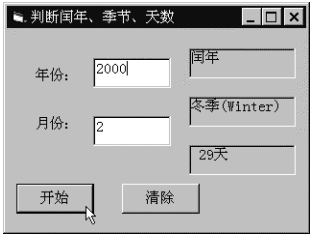


图 5-9 运行结果

### 5.3 计时器控件

计时器 (Timer) 控件能有规律地以一定的时间间隔激发计时器事件而执行相应的程序代码。计时器控件在设计时显示为一个小时钟图标，而在运行时并不显示此图标，通常用标签来显示时间。

#### 1. 计时器控件的主要属性

计时器控件的主要属性有两个。

##### (1) Enabled 属性

该属性为真时，定时器开始工作；为假时，暂停。

(2) Interval 属性

Interval（事件间隔）属性是一个非常重要的属性，表示两个计时器事件之间的时间间隔，其值以 ms 为单位，介于 0~64 767ms 之间，所以最大的时间间隔约为 1.5min。

- 如果需要屏蔽计时器，则将 Interval 设置为 0。
- 如果需要每 0.5s 产生一个计时器事件，则将 Interval 属性值设置为 500，这样，每隔 500ms（即 0.5s）就激发一次计时器事件，从而执行相应的 Interval 事件过程。
- 如果需要每 1s 产生一个计时器事件，则将 Interval 属性值设置为 1000，这样，每隔 1000ms（即 1s）激发一次计时器事件。

2. 计时器控件使用示例

【例 5-13】 设计数字计时器。

设计步骤如下。

- (1) 建立用户界面。在窗体上建立一个计时器控件和两个标签控件，如图 5-10 所示。
- (2) 设置对象属性，见表 5-3。
- (3) 编写程序代码。

编写计时器控件 Timer1 的 Timer（计时器）事件代码：

```
Private Sub Timer1_Timer()  
    Label2.Caption = Time$           ' 在标签上显示当前时间  
End Sub
```

表 5-3 属性设置

对 象	属 性	属 性 值
Form	Caption	数字计时器
Label1	Caption	当前时间为:
Label2	BackColor	(白色)
	BordeStyle	1 - Fixed Single
	Caption	
Timer1	Interval	1000

运行程序，结果如图 5-11 所示。

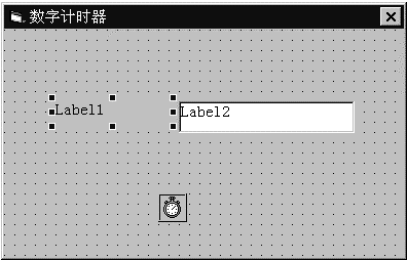


图 5-10 建立用户界面



图 5-11 程序运行结果

【例 5-14】 在窗体上设计一个能按 12 小时格式和 24 小时格式进行转换的数字时钟。

【分析】 本题需要使用的函数有 3 个。

Time 函数：返回系统当前时间。

Hour 函数：返回时间表达式中的小时数。

Format(Time, Form1.Tag)函数：按照指定格式 Form1.Tag = "hh:mm:ss AM/PM"（12 小时格式）或 Form1.Tag = "hh:mm:ss"（24 小时格式）返回系统当前时间。

设计步骤如下。

（1）建立用户界面。新建一个工程，进入窗体设计器，增加一个命令按钮Command1、一个计时器控件 Timer1 和两个标签 Label1, Label2, 参见图 5-12(a)。其中，计时器控件 Timer1 可以放在窗体的任何位置。

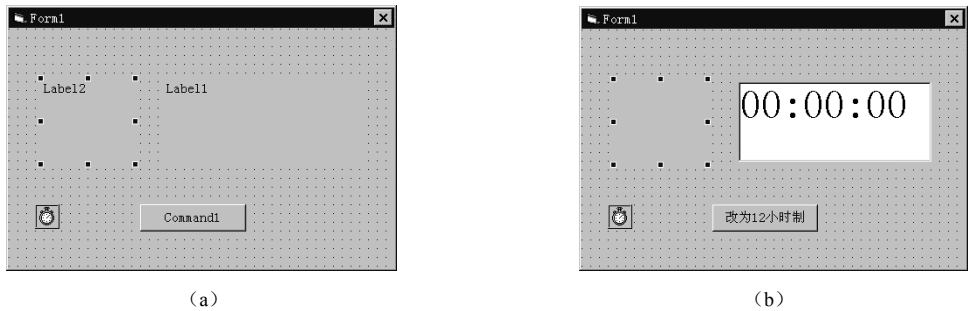


图 5-12 建立界面与设置属性

（2）设置对象属性，见表 5-4。

表 5-4 属性设置

对 象	属 性	属 性 值
Form1	Tag	hh:mm:ss AM/PM
Label1	Caption	00:00:00
	BackColor	（白色）
	BordeStyle	1-Fixed Single
Label2	Caption	
	Visible	False
Command1	Caption	改为 12 小时制
Timer1	Interval	1000

其余对象属性，参见图 5-12（b）。

（3）编写事件代码。

编写计时器控件 Timer1 的 Timer（计时器）事件代码：

```
Private Sub Timer1_Timer()  
    Label1.Caption = Format(Time, Form1.Tag)  
    If Hour(Time) > 12 Then  
        Label2.Caption = "下午"  
    Else  
        Label2.Caption = "上午"  
    End If  
End Sub
```

编写“改为 12 小时制”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    If Command1.Caption = "改为 12 小时制" Then  
        Form1.Tag = "hh:mm:ss AM/PM"  
        Command1.Caption = "改为 24 小时制"  
        Label2.Visible = True  
    Else  
        Form1.Tag = "hh:mm:ss"  
        Command1.Caption = "改为 12 小时制"  
        Label2.Visible = False  
    End If  
End Sub
```

运行程序，结果如图 5-13 所示。

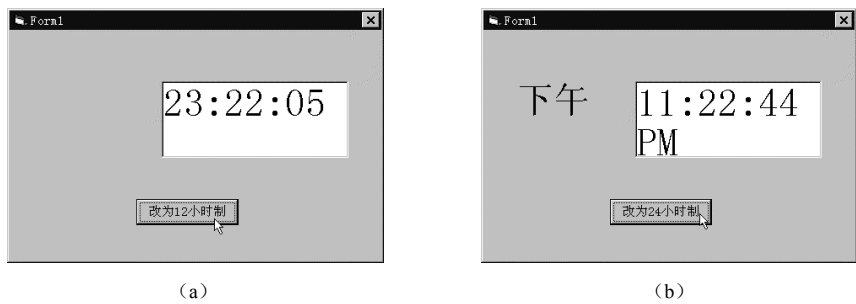




图 5-13 数字时钟

## 5.4 单选钮和复选框

大多数应用程序需要向用户提供选择，如简单的“是/否”选项等。VB 提供的用于选择的标准控件有：单选钮（OptionButton）、复选框（CheckBox）、列表框（ListBox）、组合框（ComboBox），本节只介绍单选钮和复选框。

### 5.4.1 单选钮控件

单选钮的左边有一个  图标。一般来说，单选钮总是成组（单选钮组）出现的，用户在一组单选钮中必须选择一项，并且最多只能选择一项。当某一项被选定后，其左边的圆圈中出现一个黑点 。单选钮主要用于在多个选项中由用户选择其中一个选项的情况。

#### 1. 单选钮控件的主要属性

单选钮控件的主要属性，有以下 4 个。

##### （1）Alignment 属性

- 若 Alignment 属性值为 0，表示单选钮在左边，标题显示在右边。这是默认设置。
- 若 Alignment 属性值为 1，表示单选钮在右边，标题显示在左边。



(2) Value 属性

- 若 Value 属性值为真，表示单选钮被选定。如果要使某个按钮成为单选钮组中的默认按钮，就要在设计时将其 Value 属性设置成真。这个按钮将保持被选中状态，直到用户选择另一个不同的按钮或用代码改变它。
- 若 Value 属性值为假，表示单选钮未被选定。这是默认设置。

(3) Style 属性

- Style 属性值为 0-Standard 时，是标准方式。
- Style 属性值为 1-Graphical 时，是图形方式。

(4) Enabled 属性

要禁用某个单选钮，可将其 Enabled 属性设置为假。程序运行时，此单选钮显示为浅灰色，表示无法选取。

2. 选用单选钮的方法

选择某个单选钮可以用下面方法之一：

- 在运行期间，单击选中单选钮；
- 用 Tab 键定位到单选钮组，然后在组内使用方向键定位单选钮；
- 用代码将单选钮的 Value 属性设置为真（即 Option1.Value = True）；
- 使用在单选钮标题中指定的快捷键。

3. 单选钮的事件

单选钮和复选框都可以接收 Click 事件，但一般不需要编写 Click 事件代码。因为当用户单击单选钮或复选框时，它们会自动改变状态。

4. 使用单选钮组

单选钮的一个特点是当选定其中一个后，其余就自动关闭。但当需要在同一窗体中建立几组相互独立的单选钮时，就要用框架（Frame）将每一组单选钮框起来，这样在一个框架内的单选钮为一组，它们的操作不影响框外其他组的单选钮。

5. 单选钮应用示例

**【例 5-15】** 修改例 5-14，使用单选钮组，在窗体上设计一个能按 12 小时格式和 24 小时格式进行转换的数字时钟。

设计步骤如下。

(1) 修改用户界面并设置对象属性。

在例 5-14 中，删去命令按钮 Command1，增加一个框架 Frame1。选定框架后，在其中增加两个单选钮 Option1 和 Option2。新增控件的属性设置，见表 5-5。

表 5-5 属性设置

对 象	属 性	属 性 值
Frame1	Caption	改变时间格式
Option1	Caption	12 小时
Option2	Caption	24 小时

(2) 编写新增控件的事件代码。

原例 5-14 中计时器控件的 Timer 事件代码不变：

```
Private Sub Timer1_Timer()  
    Label1.Caption = Format(Time, Form1.Tag)  
    If Hour(Time) > 12 Then  
        Label2.Caption = "下午"  
    Else  
        Label2.Caption = "上午"  
    End If  
End Sub
```

编写“12 小时”单选钮 Option1 的 Click 事件代码：

```
Private Sub Option1_Click()  
    Form1.Tag = "hh:mm:ss AM/PM"  
    Label2.Visible = True  
End Sub
```

编写“24 小时”单选钮 Option2 的 Click 事件代码：

```
Private Sub Option2_Click()  
    Form1.Tag = "hh:mm:ss"  
    Label2.Visible = False  
End Sub
```

运行程序，结果如图 5-14 所示。

【例 5-16】 输入圆的半径  $r$ ，利用单选钮，选择计算面积、周长或面积与周长等。设计步骤如下。

(1) 建立用户界面，参见图 5-15。设置对象属性，见表 5-6。



图 5-14 使用单选按钮

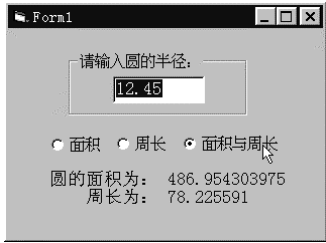


图 5-15 选择计算圆面积、圆周长

表 5-6 属性设置

对 象	属 性	属 性 值	说 明
Frame1	Caption	请输入圆的半径:	
Option1	Caption	面积	
	Value	True	按钮被选中
Option2	Caption	周长	
Option3	Caption	面积与周长	

(2) 编写代码。

基本的代码是文本框 Text1 的 KeyPress 事件代码：

```
Private Sub Text1_KeyPress(KeyAscii As Integer)  
    Dim r As String  
    If KeyAscii = 13 Then  
        pi = 3.14159  
        r = Val(Text1.Text)  
        Select Case True  
            Case Option1.Value  
                n = pi * r * r  
                Label1.Caption = "圆的面积为： " & Str(n)  
            Case Option2.Value  
                n = 2 * pi * r  
                Label1.Caption = "圆的周长为： " & Str(n)  
            Case Option3.Value  
                n = pi * r * r  
                m = 2 * pi * r  
                Label1.Caption = "圆的面积为： " & Str(n) & Chr(13) & "        周长为： " & Str(m)  
        End Select  
        Text1.SelStart = 0  
        Text1.SelLength = Len(Text1.Text)  
    End If  
End Sub
```

文本框 Text1 的 GotFocus 事件代码如下：

```
Private Sub Text1_GotFocus()  
    Text1.SelStart = 0  
    Text1.SelLength = Len(Text1.Text)  
End Sub
```

3 个单选钮具有相同的 Click 事件代码：

```
Private Sub Option1_Click()  
    Text1.SetFocus  
End Sub  
Private Sub Option2_Click()  
    Text1.SetFocus  
End Sub  
Private Sub Option3_Click()  
    Text1.SetFocus  
End Sub
```

【例 5-17】 设计滚动字幕，使“欢迎使用学生成绩管理系统”字样自右至左地反复移动。

设计步骤如下。

(1) 建立用户界面。新建一个工程，进入窗体设计器，增加一个计时器控件 Timer1、一个标签控件 Label1 和一个命令按钮 Command1，4 个单选钮 Option1~Option4。其中，计时器控件 Timer1 可以放在窗体的任何位置。

(2) 设置对象属性。修改 Timer1 的属性：Interval 改为 100，Enabled 改为 False。修改 Option1~Option4 的 Style 属性为：1-Graphical（图形方式）。其他属性修改如图 5-16 所示。

(3) 编写事件代码。

编写“开始”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    If Command1.Caption = "&S 暂停" Then  
        Command1.Caption = "&C 继续"  
        Timer1.Enabled = False  
    Else  
        Command1.Caption = "&S 暂停"  
        Timer1.Enabled = True  
    End If  
End Sub
```

通过在不断激发的 Timer 事件中改变标签的 Left 属性，从而改变标签的位置。编写计时器控件 Timer1 的 Timer 事件代码：

```
Private Sub Timer1_Timer()  
    If Label1.Left + Label1.Width > 0 Then  
        Label1.Move Label1.Left - 20      ' 当标签右边位置大于 0 时，标签向左移  
    Else  
        Label1.Left = Form1.ScaleWidth    ' 否则标签从头开始  
    End If  
End Sub
```

依次编写单选钮 Option1~Option4 的 Click 事件代码：

```
Private Sub Option1_Click()  
    Label1.FontName = "宋体"  
End Sub  
Private Sub Option2_Click()  
    Label1.FontName = "隶书"  
End Sub  
Private Sub Option3_Click()  
    Label1.FontName = "黑体"  
End Sub  
Private Sub Option4_Click()  
    Label1.FontName = "楷体_GB2312"  
End Sub
```

运行程序，结果如图 5-16 所示。

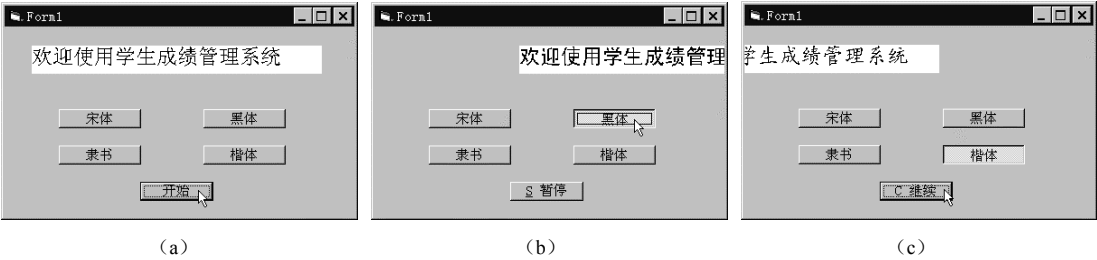


图 5-16 滚动字幕程序运行结果

### 5.4.2 复选框控件

复选框（CheckBox）的左边有一个 ☐ 图标。使用复选框列出可供用户选择的选项，用户根据需要选定其中的一项或多项。当某一项被选中后，其左边的小方框中就多一个对号 ☒。

#### 1. 复选框控件的常用属性

复选框的常用属性与单选按钮基本相同，如复选框的 **Caption** 属性可以指定出现在复选框旁边的文本，而 **Picture** 属性用来指定当复选框被设计成图形按钮时的图像，但 **Value** 属性与单选按钮不同。复选框的状态属性 **Value** 取值如下。

- 0 - Unchecked: 复选框未被选定，默认设置。
- 1 - Checked: 复选框被选定。
- 2 - Grayed: 复选框变成灰色，禁止用户选择。

#### 2. 复选框应用示例

**【例 5-18】** 修改例 5-17，增加两个复选框，从而控制滚动字幕内容是否加下划线和显示为斜体。

设计步骤如下。

（1）建立用户界面，设置对象属性。在例 5-17 的基础上，增加两个复选框。控件的属性设置，如图 5-17 所示。

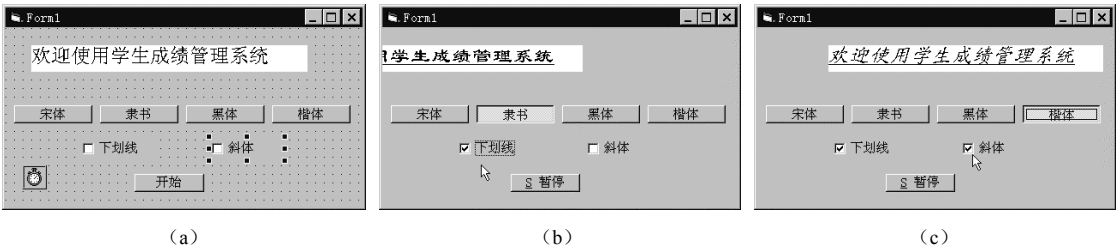


图 5-17 复选框应用示例

（2）编写事件代码。

其他事件代码不变，增加下面的事件代码：

```
Private Sub Check1_Click()  
    If Check1.Value = 1 Then
```

```

        Label1.FontUnderline = True
    Else
        Label1.FontUnderline = False
    End If
End Sub
Private Sub Check2_Click()
    If Check2.Value = 1 Then
        Label1.FontItalic = True
    Else
        Label1.FontItalic = False
    End If
End Sub

```

运行程序，结果如图 5-17 所示。

## 习题 5

### 一、选择题

5.1 下列程序段的执行结果为 ( )。

```

X=2
Y=5
If X * Y < 1 Then Y=Y-1 Else Y=-1
Print Y-X>0

```

A) True                      B) False                      C) -1                      D) 1

5.2 下面语句正确的是 ( )。

A) If  $x < 3 * y$  And  $x > y$  Then  $y = x^3$                       B) If  $x < 3 * y$  And  $x > y$  Then  $y = 3x$   
 C) If  $x < 3 * y : x > y$  Then  $y = x^3$                       D) If  $x < 3 * y : x > y$  Then  $y = x ** 3$

5.3 下列程序段执行结果为 ( )。

```

x=5
y=-6
If Not x>0 Then x=y-3 Else y=x+3
Print x-y;y-x

```

A) -3    3                      B) 5    -9                      C) 3    -3                      D) -6    5

5.4 下列语句正确的是 ( )。

A) If  $A \neq B$  Then Print "A 不等于 B"  
 B) If  $A < > B$  Then Printf "A 不等于 B"  
 C) If  $A < > B$  Then Print "A 不等于 B"  
 D) If  $A \neq B$  Print "A 不等于 B"

5.5 计算 z 的值，当 x 大于 y 时， $z=x$ ；否则  $z=y$ 。下列语句错误的是 ( )。

A) If  $x > y$  Then  $z=x : z=y$                       B) If  $x > y$  Then  $z=x$  Else  $z=y$

C)  $z=y$ :If  $x>=y$  Then  $z=x$

D) If  $x<=y$  Then  $z=y$  Else  $z=x$

5.6 下列程序段的执行结果为 ( )。

```
a=95
If a>60 Then I=1
If a>70 Then I=2
If a>80 Then I=3
If a>90 Then I=4
Print "I="; I
```

A) I=1

B) I=2

C) I=3

D) I=4

5.7 下面程序段执行结果为 ( )。

```
x=Int(Rnd()+4)
Select Case x
    Case 5
        Print "excellent"
    Case 4
        Print "good"
    Case 3
        Print "pass"
    Case Else
        Print "fail"
End Select
```

A) excellent

B) good

C) pass

D) fail

5.8 以下程序段运行时,若从键盘输入字符“-”,则输出结果为 ( )。

```
op$=InputBox("op=")
If op$="+" Then a=a+2
If op$="-" Then a=a-2
Print a
```

A) 2

B) -2

C) 0

D) +2

5.9 在窗体上画一个名称为 Timer1 的计时器控件,要求每隔 0.5 秒发生一次计时器事件,则以下正确的属性设置语句是 ( )。

A) Timer1.Interval=0.5

B) Timer1.Interval=5

C) Timer1.Interval=50

D) Timer1.Interval=500

## 二、填空题

5.10 有下面一个程序段,从文本框中输入数据,如果该数据满足条件:除以 4 余 1 且除以 5 余 2,则输出;否则,将焦点定位在文本框中,并清除文本框的内容。请补全程序。

```
Private Sub Command1_Click()
    x=Val(Text1.Text)
    If ____ Then
```

```

        Print x
    Else
        Text1.Text=""
    _____
End If
End Sub

```

5.11 给定年份，下列程序用来判断该年是否是闰年。请补全程序。

```

Sub YN()
    Dim x As Integer
    x=InputBox("请输入年号")
    If (x Mod 4=0 _____ x Mod 100 <> 0) _____ (x Mod 400=0) Then
        Print "是闰年"
    Else
        Print "不是闰年"
    End If
End Sub

```

5.12 下列程序的功能是：当  $x < 50$  时， $y = 0.8x$ ；当  $50 \leq x \leq 100$  时， $y = 0.7x$ ；当  $x > 100$  时，没有意义。请补全程序。

```

Private Sub Command1_Click()
    Dim x As Single
    x=InputBox("输入 x 的值")

    _____
    Case Is<50
        y=0.8 * x
    Case 50 To 100
        y=0.7 * x
    _____
    Print "输入的数据出界！"
End Select
Print x , y
End Sub

```

5.13 在窗体上画一个文本框和一个计时器控件，名称分别为 Text1 和 Timer1，在属性窗中把计时器的 Interval 属性设置为 100，Enabled 属性设置为 False。程序运行后，如果单击命令按钮，则每隔 1 秒在文本框中显示一次当前的时间。请补全程序。

```

Private Sub Command1_Click()
    Timer1._____
End Sub
Private Sub Timer1_Timer()
    Text1.Text=Time
End Sub

```



### 三、编程题

5.14 编写计算铁路运费的程序。假设铁路托运行李，规定每张客票托运费计算方法是：行李重不超过 50 千克时，每千克 0.25 元；超过 50 千克而不超过 100 千克时，其超过部分每千克 0.35 元；超过 100 千克时，其超过部分每千克 0.45 元。要求输入行李重量，可计算并输出托运的费用。

5.15 若基本工资大于等于 600 元，增加 20%工资；若小于 600 元，且大于等于 400 元，则增加 15%工资；若小于 400 元，则增加 10%工资。请根据用户输入的基本工资，计算出增加后的工资。

5.16 求一元二次方程  $ax^2+bx+c=0$  的根。

5.17 编写一个对输入字符进行大、小写转换的程序。转换规则为：将其中的大写字母转换成小写字母，小写字母转换成大写字母，空格不转换，其余转换成“\*”。要求每输入一个字符后，马上就进行判断和转换。

5.18 设计个人资料输入界面，使用单选按钮组输入性别与民族，用复选框输入个人爱好。

5.19 利用单选按钮组控制输入文本的字体。

5.20 文本框的 PasswordChar 属性可以隐蔽用户通过键盘输入的字符。编写程序，利用文本框检查用户口令。

5.21 设计一个计时器，能够设置倒计时的时间，并进行倒计时。

5.22 利用图形复选框来控制文本的字体风格。

## 第 6 章 循环结构程序设计

在编程中常常遇到这样的情况：某一类问题的计算和处理方法完全一样，只是要求重复计算多次，而每次使用的数据都是按照一定的规律变化的。例如，需要对一个班 30 名学生的成绩进行检查，将不及格者打印出来。类似这样的问题，就要用到循环结构。

程序设计中的循环结构（简称循环）是指在程序中从某处开始有规律地反复执行某一操作块（或程序块）的现象。被重复执行的操作块（或程序块）称为循环体，循环体的执行与否及次数多少视循环类型与条件而定。当然，无论何种类型的循环结构，其共同的特点是：必须确保循环体的重复执行能被终止（即非无限循环）。

VB 中常用的循环语句有 For...Next 语句和 Do...Loop 语句。For...Next 循环用于已知循环次数的情况，而 Do...Loop 循环主要用于不知道循环次数的情况，在给定的条件满足时执行循环体。

### 6.1 For...Next 循环语句

For...Next 循环语句按指定次数执行循环体，它在循环体中使用一个循环变量（计数器），每重复一次循环后，循环变量的值就会自动增加或者减少。

#### 1. For...Next 语句的语法格式

For...Next 语句的语法格式为：

```
For <循环变量> = <初值> To <终值> [Step <步长> ]  
    [ <语句组 1> ]  
    [ Exit For ]  
    [ <语句组 2> ]  
Next [ <循环变量> ]
```

#### 【说明】

- ① <循环变量> 为必要参数，是用做循环计数器的数值变量。该变量不能是数组元素。
- ② <初值> 和 <终值> 都是必要参数。<步长> 可以是正数或负数。当步长的值为 1 时，可以省略。
- ③ 可以省略 Next 语句中的 <循环变量>，但写上 <循环变量> 将提高程序的可读性。

#### 2. For...Next 语句的执行过程

进入 For...Next 循环后，首先把 <初值> 赋给 <循环变量>，检查 <循环变量> 的值是否超过 <终值>。如果超过就停止执行循环体，跳出循环，执行 Next 后面的语句；否则执行一次循环体，然后把 <循环变量> + <步长> 的值赋给 <循环变量>，重复上述过程。

这里所说的“超过”有两种含义，即大于或小于：

- 当〈步长〉为正值时，检查〈循环变量〉的值是否大于〈终值〉；
  - 当〈步长〉为负值时，检查〈循环变量〉的值是否小于〈终值〉。
- 可以在循环中的任何位置放置任意个 Exit For 语句，随时退出循环。

### 3. For...Next 循环的循环次数

For...Next 循环遵循“先检查，后执行”的原则，即先检查〈循环变量〉是否超过〈终值〉，然后决定是否执行循环体。因此，在下列两种情况下，循环体不被执行：

- 当〈步长〉为正数，〈初值〉大于〈终值〉时；
- 当〈步长〉为负数，〈初值〉小于〈终值〉时。

因次，循环的最少执行次数为 0 次。

当〈初值〉等于〈终值〉时，不管〈步长〉是正数还是负数，均执行一次循环体。循环次数由〈初值〉、〈终值〉和〈步长〉3 个因素决定，可以通过下式计算：

$$\text{循环次数} = \text{INT}((\text{终值} - \text{初值}) / \text{步长} + 1)$$

如果计算出的循环次数小于或者等于 0，循环次数为 0，这时系统将不执行循环体。

### 4. For...Next 语句使用示例

**【例 6-1】** 设计程序，用 For...Next 语句求  $1 + 2 + 3 + \dots + 100$  的值。

**【分析】** 采用累加的方法，用变量 s（累加器）来存放累加和（开始为 0），用变量 n 来存放加数（加到 s 中的数）。这里 n 又称为计数器，从 1 开始，到 100 结束。设计步骤如下。

- （1）建立应用程序用户界面并设置对象属性，如图 6-1 所示。
- （2）编写事件代码。编写“计算”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()
    Dim s As Integer, n As Integer
    s = 0                                ' 累加器赋初值 0
    For n = 1 To 100                     ' 初值为 1，终值为 100，步长为 1（省略）
        s = s + n                        ' 进行累加
    Next n
    Text1.Text = s                       ' 输出累加结果
End Sub
```

运行程序，结果如图 6-2 所示。

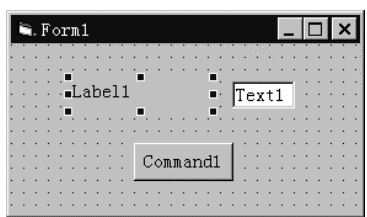


图 6-1 建立用户界面

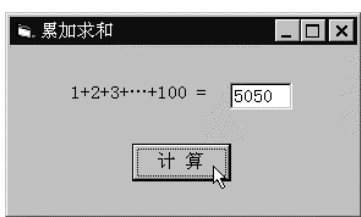


图 6-2 程序运行结果

**【例 6-2】** 输出 1000 以内所有能被 37 整除的自然数。

设计步骤如下。

(1) 建立用户界面。在新建的窗体中增加一个文本框 Text1、一个命令按钮 Command1 和一个框架 Frame1。选中 Frame1，在其中增加一个标签 Label1，用来显示程序的说明。如图 6-3 (a) 所示。

(2) 设置对象属性。设置文本框 Text1 的 MultiLine 属性值为 True，使文本框在运行时显示多行文本。设置文本框 Text1 的 ScrollBar 属性值为 2，使其具有垂直滚动条。

其他对象的属性设置如图 6-3 (b) 所示。

(3) 编写事件代码。要想使每个数据换行输出，可以用一个回车加上换行符 (Chr(13) & Chr(10)) 来产生一个行断点，把所有符合要求的数都连接到变量 a 中 (见下面程序代码)。

编写“开始”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    a = ""  
    For n = 1 To 1000  
        If n Mod 37 = 0 Then  
            a = a & Str(n) & Chr(13) & Chr(10)      ' 用 Chr(13) & Chr(10)设置行断点  
        End If  
    Next  
    Text1.Text = a  
End Sub
```

运行程序，结果如图 6-3 (c) 所示。

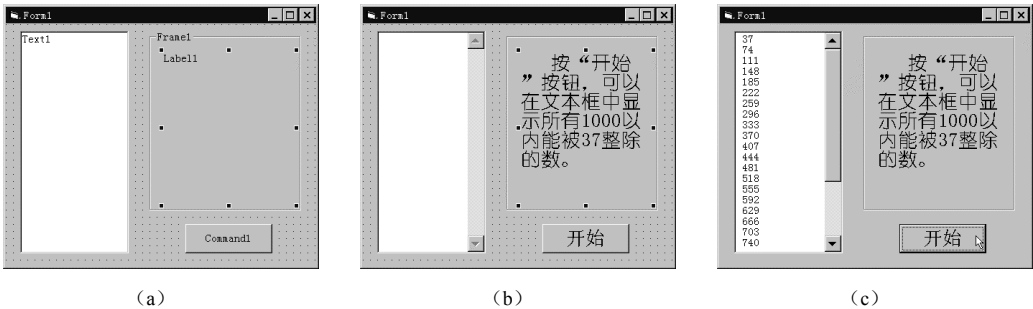


图 6-3 能被 37 整除的自然数

5. For...Next 语句的嵌套

For...Next 循环语句可以嵌套使用，嵌套层数没有具体限制，其基本要求是：

- 每个循环必须有一个唯一的变量作为控制变量；
- 内层循环必须完全放在外循环体内，内外循环不得互相交叉跨骑。

例如，下面的嵌套是错误的：

```
For a=1 To 5  
    For b=3 To 9  
        .....  
    Next a  
Next b
```

For...Next 循环的嵌套通常有以下 3 种形式。

① 一般嵌套形式。

```
For a1=...  
  For a2=...  
    For a3=...  
      .....  
    Next a3  
  Next a2  
Next a1
```

② 形式 ① 中 Next 后面的 a1, a2, a3 可以省略不写。

③ 当内层循环与外层循环有相同的终点时，可公用一个 Next 语句，但是，控制变量名不能省略。例如：

```
For a=1 To 2  
  For b=2 To 3  
    For c=3 To 4  
      Print a, b, c  
    Next c, b, a
```

【例 6-3】 打印出如图 6-4 所示的乘法“九九表”。

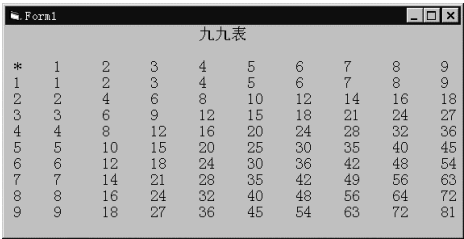


图 6-4 打印乘法“九九表”

【分析】“九九表”是一个 9 行 9 列的二维表，行和列都要变化，而且在变化中相互约束。这是一个二重循环问题。

直接在窗体上输出。窗体 Form 的 Load 事件代码为：

```
Private Sub Form_Load()  
  Show  
  FontSize = 12 ' 设置字号  
  Print Tab(25); "九九表" ' 输出标题  
  Print ' 输出空行  
  Print " * ";  
  For i = 1 To 9 ' 输出第一行数字（1~9）  
    Print Tab(i * 6); i; ' 每列空 5 格，定位输出  
  Next i  
  Print ' 换行  
  For j = 1 To 9 ' 外层循环
```

```

Print j; " ";
For k = 1 To 9          ' 内层循环
    m = j * k          ' 计算乘积
    Print Tab(k * 6); m; " "; ' 定位输出
Next k
Print                  ' 换行
Next j
End Sub

```

## 6.2 Do...Loop 循环语句

For...Next 循环总是按指定的次数执行循环体。如果事先不知道循环次数，或循环的初值和终值不明了，则需要使用 Do...Loop 语句。Do...Loop 语句有两种语法形式：

- 前测型循环结构
- 后测型循环结构

### 6.2.1 前测型 Do...Loop 循环语句

#### 1. 前测型 Do...Loop 的语法格式

前测型 Do...Loop 循环结构的循环特点是：先判断循环条件，根据条件决定是否执行循环体，执行循环体的最少次数为 0。

其语法格式为：

```

Do [{ While | Until } 〈条件〉]
    [ 〈语句组 1〉 ]
    [ Exit Do ]
    [ 〈语句组 2〉 ]

```

**Loop**

#### 【说明】

- ① 〈条件〉是条件表达式，为循环的条件，其值为 True 或 False。
- ② 〈语句组〉是一条或多条命令（循环体），当（或直到）条件为真时被重复执行。

#### 2. 前测型 Do...Loop 的执行过程

前测型 Do...Loop 循环语句先判断条件，再执行循环体。根据条件不同，可分为当型和直到型循环结构。

- 当型 Do While...Loop：当条件为真时，执行循环体；条件为假时，终止循环。
- 直到型 Do Until...Loop：当条件为假时，执行循环体；直到条件为真时，终止循环。

在 Do...Loop 循环体中，可以放置任意个数的 Exit Do 语句，随时跳出 Do...Loop 循环。Exit Do 语句通常用于条件判断之后，例如 If...Then 语句，在这种情况下，Exit Do 语句将控制权转移到紧接在 Loop 命令之后的语句。如果在嵌套的 Do...Loop 语句中使用 Exit Do，则 Exit Do 会将控制权转移到 Exit Do 所在位置的外层循环。

### 3. 前测型 Do...Loop 使用示例

【例 6-4】 用前测型 Do...Loop 语句，计算  $1 + 2 + 3 + \dots + 100$  的值。  
设计步骤如下。

- (1) 建立用户界面并设置对象属性，如图 6-5 (a) 所示。
- (2) 设置对象属性，参见图 6-5 (b)。

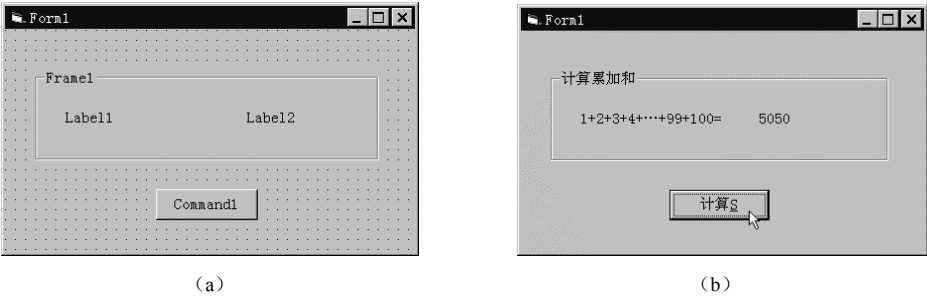


图 6-5 程序界面与运行结果

- (3) 编写事件代码。

采用当型循环结构编写“计算”命令按钮 Command1 的 Click 事件代码为：

```
Private Sub Command1_Click()  
    Dim S As Integer, n As Integer  
    S = 0 : n = 1           ' 累加器 S 赋初值 0，计数器 n 赋初值 1  
    Do While n <= 100       ' 当型循环  
        S = S + n           ' 累加和  
        n = n + 1          ' 计数器累加 1  
    Loop  
    Label2.Caption = S  
End Sub
```

“计算”命令按钮 Command1 的 Click 事件代码还可以改为直到型循环结构：

```
Private Sub Command1_Click()  
    Dim S As Integer, n As Integer  
    S = 0 : n = 1  
    Do Until n > 100        ' 直到型循环  
        S = S + n  
        n = n + 1  
    Loop  
    Label2.Caption = S  
End Sub
```

该事件代码还可以利用 Exit Do 语句来编写：

```
Private Sub Command1_Click()  
    Dim S As Integer, n As Integer  
    S = 0 : n = 1
```

```
Do
    S = S + n
    n = n + 1
    If n > 100 Then Exit Do      ' 如果 n>100，则跳出循环
Loop
Label2.Caption = S

End Sub
```

运行程序，结果如图 6-5（b）所示。

**【例 6-5】** 已知  $s = 1 \times 2 \times 3 \times \cdots \times n$ ，计算出  $s$  不大于 5000 时  $n$  的最大值。

**【分析】** 本题利用循环结构进行累乘运算。设计数器为  $n$ ，累乘器  $s = s * n$ ，其循环条件是  $s \leq 5000$ 。由于求的是最大  $n$  值，输出语句应在循环体外。

设计步骤如下。

- （1）建立用户界面并设置对象属性，如图 6-6 所示。
- （2）编写事件代码。

编写“计算”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()
    Dim n As Integer, s As Long
    CurrentY = Label2.Height + 200      ' 确定输出位置
    n = 1                                ' 计数器赋初值 1
    s = 1                                ' 累乘器赋初值 1
    Do While s <= 5000                   ' 循环条件
        n = n + 1                        ' 计数器累加 1
        s = s * n                        ' 累乘
        Print n, s                       ' 打印循环过程
    Loop
    Label1.Caption = "n = " & Str(n - 1)

End Sub
```

运行程序，结果如图 6-6 所示。

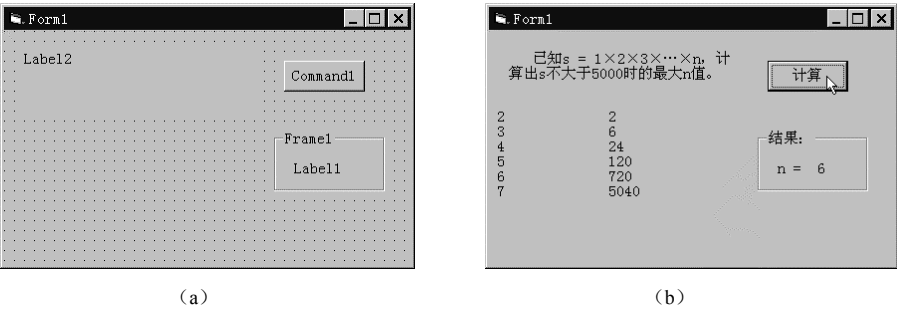


图 6-6 程序界面与运行结果

**【例 6-6】** 输入一个正整数，利用 Do 循环判断是否为素数。

**【分析】** 所谓素数，是指除了 1 和该数本身外，不能被任何整数整除的数。判断一个自



然数  $n$  ( $n \geq 3$ ) 是否为素数, 只要依次用  $2 \sim \sqrt{n}$  之间的整数作为除数去除  $n$ , 若  $n$  不能被其中任何一个数整除, 则  $n$  即为素数。

设计步骤如下。

(1) 建立用户界面并设置对象属性, 如图 6-7 所示。

(2) 编写事件代码。编写“判定素数”命令按钮 Command1 的 Click 事件代码:

```
Private Sub Command1_Click()
```

```
Dim n As Long
```

```
Select Case Val(Text1.Text)
```

```
Case Is < 3
```

```
MsgBox "请输入一个大于 2 的整数!", vbInformation + vbOKOnly, "注意"
```

```
Case Is > 2147483647
```

```
MsgBox "此数太大!", vbInformation + vbOKOnly, "注意"
```

```
Case Else
```

```
n = Val(Text1.Text)
```

```
s = 0: i = 2
```

```
Do While i <= Sqr(n) And s = 0
```

```
If n Mod i = 0 Then
```

```
s = 1
```

```
Else
```

```
i = i + 1
```

```
End If
```

```
Loop
```

```
If s = 0 Then
```

```
a = "是一个素数"
```

```
Else
```

```
a = "不是素数"
```

```
End If
```

```
Label1.Caption = Str(n) & a
```

```
End Select
```

```
Text1.SetFocus
```

```
End Sub
```

运行程序, 结果如图 6-7 所示。

**【例 6-7】** 输出 100~200 之间不能被 3 整除的数。

**【分析】** 可以用多种循环语句来实现。根据题意, 某数不能被 3 整除, 可以用 Mod 运算来完成, 即用  $x \text{ Mod } 3 \neq 0$  来表示  $x$  不能被 3 整除。计数器从 100 开始, 到 200 结束。

设计步骤如下。

(1) 建立用户界面并设置对象属性, 如图 6-8 (a) 所示。

(2) 编写事件代码。

编写“开始”命令按钮 Command1 的 Click 事件代码:

```
Private Sub Command1_Click()  
    Dim x As Integer  
    x = 100  
    Do Until x > 200  
        If x Mod 3 <> 0 Then  
            Text1.Text = Text1.Text & Str(x) & Chr(13) & Chr(10)  
        End If  
        x = x + 1  
    Loop  
End Sub
```

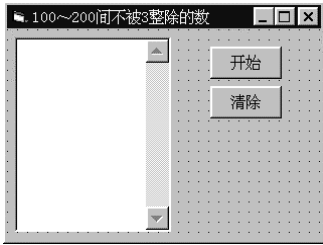
编写“清除”命令按钮 Command2 的 Click 事件代码：

```
Private Sub Command2_Click()  
    Text1.Text = ""  
End Sub
```

运行程序，结果如图 6-8（b）所示。



图 6-7 判断素数



(a)



(b)

图 6-8 用户界面与程序运行结果

6.2.2 后测型 Do…Loop 循环语句

后测型 Do…Loop 循环结构的执行特点是：先执行循环体，然后判断条件，根据条件决定是否继续执行循环，因此执行循环的最少次数为 1。

1. 后测型 Do…Loop 的语法格式

后测型 Do…Loop 循环结构的语法格式为：

```
Do  
    [ <语句组 1> ]  
    [ Exit Do ]  
    [ <语句组 2> ]  
Loop [{While | Until} <条件> ]
```

【说明】

- ① <条件> 是条件表达式，为循环的条件，其值为 True 或 False。
- ② <语句组> 是一条或多条命令（循环体），当（或直到）条件为真时，被重复执行。

## 2. 后测型 Do...Loop 的执行过程

后测型 Do...Loop 是先执行一次循环体后，再进行条件判断，分为当型和直到型两种：

- 当型 Do...While Loop：当条件为真时，继续执行循环体；若条件为假，则终止循环。
- 直到型 Do...Until Loop：当条件为假时，继续执行循环体；直到条件为真时，终止循环。

在 Do...Loop 循环体中，可以放置任意个 Exit Do 语句，随时跳出 Do...Loop 循环。

## 3. 后测型 Do...Loop 使用示例

**【例 6-8】** 输入有效数字的位数，利用下述公式计算圆周率 $\pi$ 的近似值：

$$\pi = 2 \cdot \frac{2}{\sqrt{2}} \cdot \frac{2}{\sqrt{2+\sqrt{2}}} \cdot \frac{2}{\sqrt{2+\sqrt{2+\sqrt{2}}}} \cdot \dots$$

**【分析】** 首先找出公式中无穷乘积各项的规律。设第  $n$  项的分母为  $p_n$ ，则第  $n+1$  项的分母为  $p_{n+1} = \sqrt{2+p_n}$ 。若设前  $n$  项乘积为  $S_n$ ，则前  $n$  项乘积为  $S_{n+1} = 2S_n / p_{n+1}$ 。

设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性，如图 6-9 所示。

(2) 编写程序代码。

根据流程图，可以写出“计算”命令按钮 Command1 的 Click 事件代码为：

```
Private Sub Command1_Click()  
    Dim m As Integer  
    m = Val(Text1.Text)  
    p = 0#: s = 2#: e = 0.1 ^ m  
    Do  
        t = s : p = Sqr(2 + p) : s = s * 2 / p  
    Loop Until Abs(t - s) < 0.1 ^ m  
    f = String(m - 1, "#")  
    Text2.Text = Format(s, "0." & f)  
    Text1.SetFocus  
End Sub
```

文本框 Text1 的 GotFocus 事件代码如下：

```
Private Sub Text1_GotFocus()  
    Text1.SelStart = 0  
    Text1.SelLength = Len(Text1.Text)  
End Sub
```

**【例 6-9】** 输入两个正整数，求它们的最大公约数。

**【分析】** 求最大公约数可以用“辗转相除法”，方法如下。

① 以大数  $m$  作为被除数，小数  $n$  作为除数，相除后余数为  $r$ 。

② 若  $r \neq 0$ ，则  $m \leftarrow n$ ， $n \leftarrow r$ ，继续相除得到新的  $r$ 。若仍有  $r \neq 0$ ，则重复此过程，直到  $r=0$  为止。

③ 最后的  $n$  就是最大公约数。

设计步骤如下。

- (1) 建立应用程序用户界面并设置对象属性，如图 6-10 (a) 所示。
- (2) 编写程序代码。根据流程图，编写“计算”命令按钮 Command1 的 Click 事件代码为：

```
Private Sub Command1_Click()  
    Dim m As Integer, n As Integer  
    m = Val(Text1.Text)  
    n = Val(Text2.Text)  
    If m < n Then  
        t = m: m = n: n = t  
        ' 交换数据，使大数在前，小数在后  
    End If  
    Do  
        ' 求最大公约数  
        If n <= 0 Or m <= 0 Then  
            ' 检验数据范围  
            MsgBox "请重新输入数据！"  
            Exit Do  
        End If  
        r = m Mod n  
        m = n  
        n = r  
    Loop While r <> 0  
    ' 当 r<>0 时辗转相除  
    Label3.Caption = m  
    ' 输出结果  
End Sub
```

运行程序，结果如图 6-10 (b) 所示。

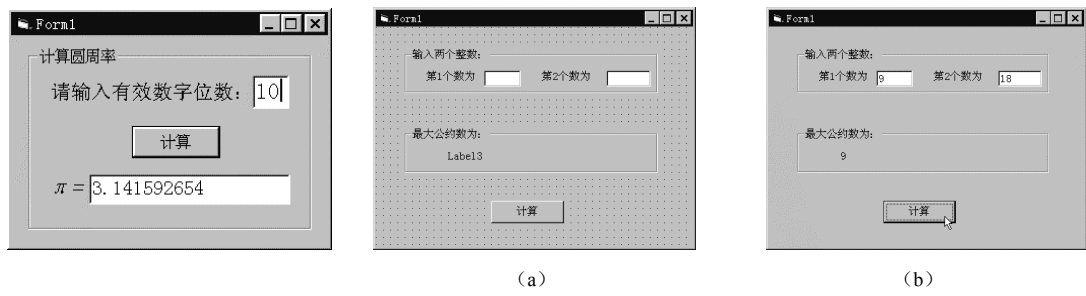


图 6-9 计算圆周率π

图 6-10 求两整数的最大公约数

**【例 6-10】** 设华氏温度为  $h$ ，摄氏温度为  $s$ ，已知将华氏温度转换为摄氏温度的公式为：

$$s = \frac{5}{9}(h - 32)$$

设计程序，实现华氏温度向摄氏温度的转换。  
本例直接在窗体上载入。窗体 Form1 的 Load 事件代码为：

```
Private Sub Form_Load()  
    Dim h As String, s As Single, ts As String
```

```

Do
    h = InputBox("请输入华氏温度", "华氏温度")      ' 利用输入对话框输入华氏温度
    If h <> "" Then
        s = Int((h - 32) * 5 / 9)                    ' 计算摄氏温度
        MsgBox "摄氏温度为" & Str(s), 0 + 48 + 256, "转换为摄氏温度"
    End If
Loop While h <> ""                                  ' 若输入框中的值不为空，反复计算

End Sub

```

运行程序，结果如图 6-11 所示。



图 6-11 华氏温度转换为摄氏温度

## 6.3 列表框与组合框

如果需要向用户提供包含一些选项和信息的列表，由用户从中进行选择，可使用列表框或组合框。列表框与组合框在使用中是不相同的。

- 列表框：任何时候都能看到多个选项。
- 组合框：平时只能看到一个选项，单击组合框右端的下拉箭头按钮可以打开具有多个选项的列表。

### 6.3.1 列表框控件

列表框（ListBox）控件通过显示多个选择项，供用户选择其中一项，达到与用户对话的目的。如果有较多的选择项，超出所画的区域而不能一次全部显示时，VB 会自动加上垂直滚动条。

#### 1. 列表框的属性

##### （1）基本属性

列表框的常用基本属性有 Name, Enabled, Visible, Index 等。

##### （2）List 属性

该属性用于设置或返回列表中的选项。该属性是一个字符型数组，存放列表框中的项目。

注意：List 数组的下标是从 0 开始的，例如 List1.List(1)表示列表框 List1 中第 2 项的值。

##### （3）Text 属性

该属性用于设置或返回列表中当前选项的文本内容。

#### (4) ListCount 属性

该属性用于返回列表框中项目的数量。ListCount -1 表示列表中最后一项的序号。

#### (5) ListIndex 属性

该属性用于返回选中的列表项序号。如果未选中任何项，则 ListIndex 的值为 -1。

#### (6) Selected 属性

该属性用于在程序运行中使用代码来选定列表中的选项，例如：List1.Selected (2) = True 表示选中了 List1 中的第 3 项，如果为 False，则表示未被选中。

#### (7) Sorted 属性

该属性用于决定列表框中项目在程序运行期间是否按字母顺序排列显示。如果 Sorted 值为 True，则项目按字母顺序排列显示；如果 Sorted 值为 False，则按项目加入的先后顺序排列显示。

#### (8) MultiSelect 属性

- 0 - None: 禁止多项选择。这时在一个列表框中只能选择一项。
- 1 - Simple: 简单多项选择。单击鼠标左键或按空格键表示选定或取消选定一个选项。
- 2 - Extended: 扩展多项选择。按下 Ctrl 键同时单击鼠标左键或按空格键，表示选定或取消选定一个选项；按下 Shift 键同时单击鼠标左键，或者按下 Shift 键并且移动方向键，就可以从前一个选定的项扩展选择到当前选项，即选定多个连续项。

## 2. 列表框的方法

列表框中的选项可以简单地在设计状态下通过 List 属性设置，也可以在程序中用 AddItem 方法来添加，用 Clear 或 RemoveItem 方法删除。

#### (1) AddItem 方法

AddItem 方法把一个项目加入列表框中，其语法格式如下：

**〈对象〉. AddItem 〈字符串表达式〉 [, 〈位置〉 ]**

##### 【说明】

- ① AddItem 方法适用于列表框或组合框。
- ② 〈字符串表达式〉是要加入列表框或组合框中的项目。
- ③ 〈位置〉决定新增项目在列表框或组合框中的位置。如果省略，则新增项目添加在最后。对于第一个项目，位置为 0。

#### (2) Clear 方法

Clear 方法可清除列表框的所有内容，其语法格式如下：

**〈对象〉. Clear**

**【说明】** Clear 方法中的对象可以是列表框、组合框或剪贴板。

#### (3) RemoveItem 方法

RemoveItem 方法可以从列表框中除去一个项目，其语法格式如下：

**〈对象〉. RemoveItem 〈位置〉**

**【说明】** 该方法适用于列表框或组合框。〈位置〉是被删除项目在列表框或组合框中的位置。对于第一个项目，位置为 0。

### 3. 列表框使用示例

【例 6-11】 在列表框中，显示 200 以内能被 6 整除的自然数。

【分析】若某数  $n$  能被 6 整除，即  $n \bmod 6 = 0$ 。

设计步骤如下。

(1) 建立用户界面并设置对象属性。在新建的窗体中增加一个列表框 List1，一个标签 Label1 和两个命令按钮 Command1、Command2。其中各对象的属性设置，如图 6-12 (a) 所示。

(2) 编写事件代码。

编写“显示”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    List1.Clear ' 清空列表框的内容  
    For n = 1 To 200  
        If n Mod 6 = 0 Then List1.AddItem n ' 若 n 能被 6 整除，则添加到列表框中  
    Next n  
End Sub
```

编写“关闭”命令按钮 Command2 的 Click 事件代码：

```
Private Sub Command2_Click()  
    Unload Me  
End Sub
```

运行程序，结果如图 6-12 (b) 所示。

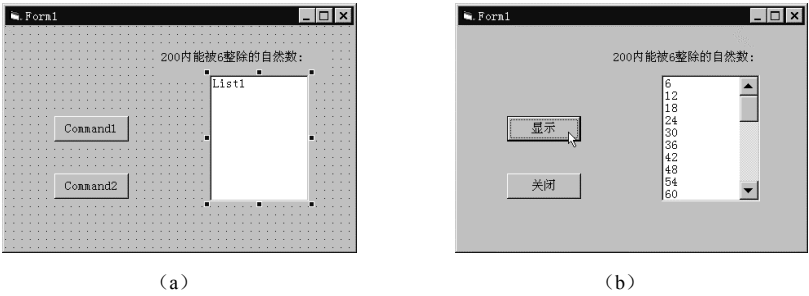


图 6-12 200 内能被 6 整除的自然数

【例 6-12】 从文本框中输入或从列表框中选择姓名，并显示出来，如图 6-13 所示。

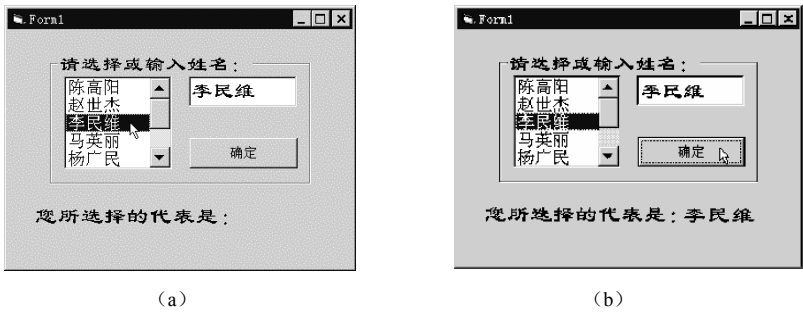


图 6-13 从文本框中输入或从列表框中选择姓名

设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性。新建一个工程，进入窗体设计器，增加一个框架 Frame1 和一个标签 Label1。激活 Frame1 后，在其中增加一个列表框 List1，一个文本框 text1 和一个命令按钮 Command1。修改对象属性参见表 6-1。

表 6-1 属性设置

对 象	属 性	属 性 值	说 明
Frame1	Caption	请选择或输入姓名:	
Label1	Caption	您所选择的代表是:	
Text1	Text		清空
Command1	Caption	确定	

设置列表框 List1 的属性，在其中依次加入：“陈高阳”、“赵世杰”、“李民维”、“马英丽”、“杨广民”、“李灵君”、“陈吉至”（每输完一项，按 Ctrl+Enter 键可输入下一项）。

(2) 编写程序代码。

编写列表框 List1 的 Click 事件代码：

```
Private Sub List1_Click()  
    Text1.Text = List1.Text  
End Sub
```

编写文本框 Text1 的 Change 事件代码：

```
Private Sub Text1_Change()  
    Label1.Caption = "您所选择的代表是: "  
End Sub
```

编写命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    Text1.SelStart = 0  
    Text1.SelLength = Len(Text1.Text)  
    Label1.Caption = "您所选择的代表是: " + Text1.Text  
End Sub
```

在上面的例子中，列表框中的各项数据是设计时在属性窗口中设置的。下面的例子说明如何在程序运行中向列表框添加新的项目，或移去列表框中的选定项。

【例 6-13】 在列表框之间移动数据，如图 6-14 所示。

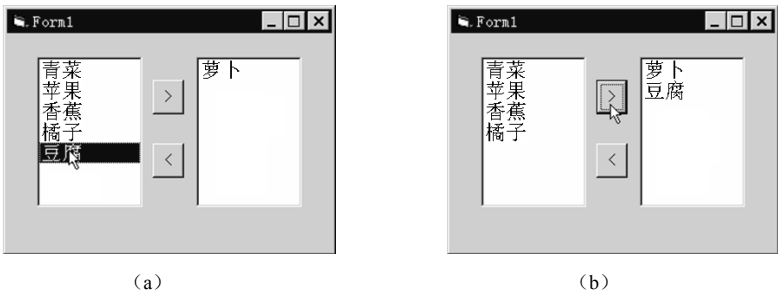


图 6-14 在列表框中移动数据



设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性。新建一个工程，进入窗体设计器，增加两个列表框 List1, List2 和两个命令按钮 Command1, Command2。然后设置对象属性见表 6-2。

表 6-2 属性设置

对 象	属 性	属 性 值	说 明
Command1	Caption	>	“>” 按钮
Command2	Caption	<	“<” 按钮

(2) 编写程序代码。

编写窗体的 Load 事件代码：

```
Private Sub Form_Load()  
    List1.AddItem "青菜"  
    List1.AddItem "萝卜"  
    List1.AddItem "豆腐"  
    List1.AddItem "苹果"  
    List1.AddItem "香蕉"  
    List1.AddItem "橘子"
```

End Sub

编写命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    List2.AddItem List1.Text  
    List1.RemoveItem List1.ListIndex
```

End Sub

编写命令按钮 Command2 的 Click 事件代码：

```
Private Sub Command2_Click()  
    List1.AddItem List2.Text  
    List2.RemoveItem List2.ListIndex
```

End Sub

另外，列表框还允许选择多个选项。

【例 6-14】 修改例6-13，允许从一个列表框中将选中的多个选项移至另一个列表框中，如图 6-15 所示。

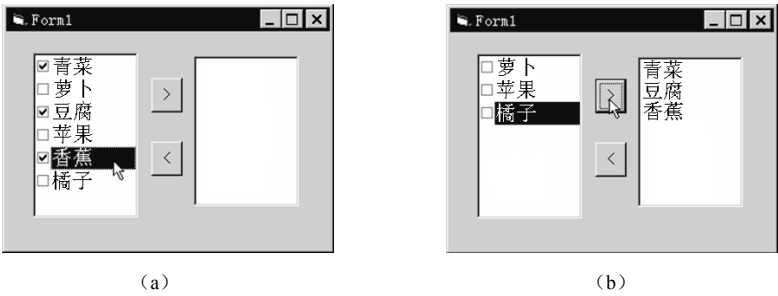


图 6-15 双列表框

(1) 修改对象属性参见表 6-3。

表 6-3 属性设置

对 象	属 性	属 性 值	说 明
List1	Style	1 - Checkbox	具有复选框风格，可以复选
List2	MultiSelect	2 - Extended	允许多项复选

(2) 修改程序代码。

命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    i = 0  
    Do While i < List1.ListCount - 1  
        If List1.Selected(i) = True Then  
            List2.AddItem List1.List(i)  
            List1.RemoveItem i  
        Else  
            i = i + 1  
        End If  
    Loop  
End Sub
```

命令按钮 Command2 的 Click 事件代码：

```
Private Sub Command2_Click()  
    i = 0  
    Do While i < List2.ListCount - 1  
        If List2.Selected(i) = True Then  
            List1.AddItem List2.List(i)  
            List2.RemoveItem i  
        Else  
            i = i + 1  
        End If  
    Loop  
End Sub
```

6.3.2 组合框控件

组合框（ComboBox）是组合了列表框和文本框的特性而形成的一种控件。组合框是一种独立的控件，它兼有列表框和文本框的功能，可以像列表框一样，让用户通过鼠标选择所需的项目；也可以像文本框一样，用输入的方式选择项目，但输入的内容不能自动添加到列表框中。若用户选中列表框中的某项，该项内容自动装入文本框中。组合框比列表框占用的屏幕空间要小。

列表框的属性基本上都可用于组合框，此外组合框还有一些自己的属性。

组合框有 3 种不同风格，即下拉组合框、简单组合框和下拉列表框。组合框的风格由 Style 属性值决定，其值分别为 0, 1, 2。3 种组合框如图 6-16 所示。



图 6-16 Style 属性的使用

### 1. 下拉组合框

下拉组合框的 Style 属性值为 0（默认），显示在屏幕上的仅是文本编辑框和一个下拉箭头按钮。执行时，用户可用键盘直接在文本框中输入内容，也可单击右边的下拉箭头按钮，打开列表框进行选择，选中的内容将显示在文本框中。

这种组合框允许用户输入不属于列表内的选项。当用户再次单击下拉箭头按钮时，下拉出来的列表就会消失，仅显示文本框。

### 2. 简单组合框

简单组合框的 Style 属性值为 1。它列出了所有的项目供用户选择，右边没有下拉箭头按钮，列表框不能被收起和拉下，与文本框一起显示在屏幕上。可以在文本框中用键盘输入列表框中没有的选项。

注意，必须用鼠标拖动滚动条才能显示全部项目。

### 3. 下拉列表框

下拉列表框的 Style 属性值为 2。其功能与下拉组合框类似，区别是，在下拉列表框中不能输入列表框中没有的选项。

### 4. 组合框使用示例

#### 【例 6-15】 利用组合框设计“自动抽奖机”。

设计步骤如下。

（1）建立应用程序用户界面并设置对象属性。新建一个工程，进入窗体设计器，增加一个组合框 Combo1，两个标签 Label1, Label2 和两个命令按钮 Command1, Command2。将 Combo1 的 Style 属性改为 0，其他属性的设置参见图 6-17。

（2）编写事件代码。

编写组合框 Combo1 的 KeyPress 事件代码：

```
Private Sub Combo1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then                ' 按 Enter 键
        Combo1.AddItem Combo1.Text, 0    ' 接收输入的号码
        Combo1.SelStart = 0              ' 设置组合框的起始位置
        Combo1.SelLength = Len(Combo1.Text) ' 设置组合框的长度
    End If
End Sub
```

```

End If
If KeyAscii = 27 Then                                ' 按 Esc 键
    If Combo1.ListIndex <> -1 Then
        Combo1.RemoveItem Combo1.ListIndex          ' 移去选项
    End If
End If
End Sub

```



(a)



(b)

图 6-17 自动抽奖机

编写“自动抽奖”命令按钮 Command1 的 Click 事件代码，使之可以随机地抽取奖号：

```

Private Sub Command1_Click()
    Randomize
    n = Combo1.ListCount                ' 求组合框中的项目数
    a = Int(Rnd * n)                   ' 利用随机数函数求随机序号
    Combo1.ListIndex = a
    Label2.Caption = "中奖的号码是:" & Chr(13) & Combo1.Text
End Sub

```

编写“退出”命令按钮 Command2 的 Click 事件代码：

```

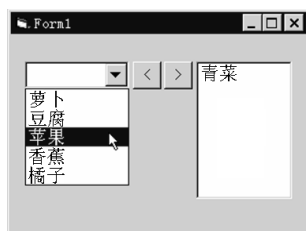
Private Sub Command2_Click()
    Unload Me
End Sub

```

运行程序，结果如图 6-17 所示。

【例 6-16】 将例 6-13 中左侧的列表框改为组合框（下拉列表框），如图 6-18 所示。

只需将左侧的列表框 List1 改为组合框 Combo1，Style 属性改为：2 - Dropdown List，并将代码中的所有 List1 改为 Combo1 即可。



(a)



(b)

图 6-18 使用下拉列表框

## 习题 6

### 一、选择题

6.1 阅读下面的程序段：

```
For a=1 To 2
    For b=1 To a
        For c=b To 2
            i=i+1
        Next
    Next
Next
Print i
```

执行上面的三重循环后，i 的值为（ ）。

- A) 4                      B) 5                      C) 6                      D) 9

6.2 设有下面的循环：

```
i=1
Do
    i=i+3
    Print i
Loop Until i> ____
```

程序运行后要执行 3 次循环体，则条件中 i 的最小值为（ ）。

- A) 6                      B) 7                      C) 8                      D) 9

6.3 下面程序段的运行结果是（ ）。

```
a=1
b=1
Do
    a=a+1
    b=b+1
Loop Until b>5
Print"k=";a;Spc(4);"b=";b+a
```

- A) k=7 b=14              B) k=6 b=6              C) k=4 b=8              D) k=6 b=12

6.4 下列程序段的执行结果为（ ）。

```
a=6
b=1
For i=1 To 3
    f=a+b
    a=b
    b=f
    Print f;
Next i
```

A) 2 3 6

B) 2 3 5

C) 2 3 4

D) 2 2 8

6.5 在窗体上画一个名称为 Command1 的命令按钮和一个名称为 Text1 的文本框，然后编写如下事件过程：

```
Private Sub Command1_Click()  
    n=Val(Text1.Text)  
    For i=2 To n  
        For j=2 To Sqr(i)  
            If i Mod j=0 Then Exit For  
        Next j  
        If j>Sqr(i)Then Print i  
    Next i  
End Sub
```

该事件过程的功能是（ ）。

A) 输出 n 以内的奇数

B) 输出 n 以内的偶数

C) 输出 n 以内的素数

D) 输出 n 以内能被 j 整除的数

6.6 下列程序段的执行结果为（ ）。

```
a=5  
For k=1 To 0  
    a=a + k  
Next k  
Print k;a
```

A) -1 6

B) -1 16

C) 1 5

D) 11 21

6.7 下列程序段的执行结果为（ ）。

```
a=3  
b=1  
For i=1 To 3  
    f=a+b  
    a=b  
    b=f  
    Print f;  
Next i
```

A) 4 3 6

B) 4 5 9

C) 6 3 4

D) 7 2 8

6.8 下列程序段的执行结果为（ ）。

```
i=9  
x=5  
Do  
    i=i+1  
    x=x+2  
Loop Until i >=7
```

```
Print "i="; i;  
Print "x="; x
```

- A) i=4 x=5                      B) i=7 x=15                      C) i=6 x=8                      D) i=10 x=7

6.9 执行下列程序段后，输出的结果是（ ）。

```
For k1=0 To 4  
    y=20  
    For k2=0 To 3  
        y=10  
        For k3=0 To 2  
            y=y + 10  
        Next k3  
    Next k2  
Next k1  
Print y
```

- A) 90                      B) 60                      C) 40                      D) 10

6.10 新建一个列表框，要实现对列表项可以复选，应设置的属性是（ ）。

- A) ScrollBars                      B) MultiSelect                      C) DataField                      D) Stretch

6.11 在窗体上画一个命令按钮，然后编写如下事件过程：

```
Private Sub Command1_Click()  
    x=0  
    Do Until x=-1  
        a=InputBox("请输入 A 的值")  
        a=Val(A)  
        b=InputBox("请输入 B 的值")  
        b=Val(B)  
        x=InputBox("请输入 x 的值")  
        x=Val(X)  
        a=a+b+x  
    Loop  
    Print a  
End Sub
```

程序运行后，单击命令按钮，依次在输入对话框中输入 5、4、3、2、1、-1，则输出结果为（ ）。

6.12 在 Visual Basic 中，组合框是文本框和（ ）特性的组合。

- A) 复选框                      B) 标签                      C) 列表框                      D) 目录列表框

6.13 下列语句中，返回列表框 List1 中项目个数的语句是（ ）。

- A) x=List1.ListCount                      B) x=ListCount  
C) x=List1.ListIndex                      D) x=ListIndex

## 二、填空题

6.14 执行下面的程序段，x 的值为\_\_\_\_\_。

```
Private Sub Command1_Click()  
    For i=1 To 9  
        a=a + i  
    Next i  
    x=Val(i)  
    MsgBox x  
End Sub
```

6.15 以下程序的功能是从键盘输入若干个学生的考试成绩，统计并输出最高分和最低分，当输入负数时结束输入，输出结果。请补全程序。

```
Dim x,amax,amin As Single  
x=InputBox("Enter a score")  
amax=x  
amin=x  
Do While ____  
    If x>amax Then  
        amax=x  
    End If  
    If ____ Then  
        amin=x  
    End If  
    x=InputBox("enter a score")  
Loop  
Print "max="; amax, "min="; amin
```

6.16 下面程序用来打印“乘法九九表”，请补全程序。

```
Dim i As Integer, j As Integer, Str1$  
Str=" "  
For i=1 To 9  
    For j=1 To 9  
        If ____ Then  
            Str1=Str1+Str$(j) + "×"+Str$(i)+ "="+Str$(Val(i * j))  
        Else  
            Str1=Str1 & Chr(13)  
            ____  
        End If  
    Next j  
Next i  
Print Str1
```



6.17 在窗体上画一个命令按钮，然后编写如下程序：

```
Function fun(By Val num As Long)As Long
```

```
    Dim k As Long
```

```
    k=1
```

```
    num=Abs(num)
```

```
    Do While num
```

```
        k=k*(num Mod 10)
```

```
        num=num \ 10
```

```
    Loop
```

```
    fun=k
```

```
End Function
```

```
Private Sub Command1_Click()
```

```
    Dim n As Long , r As Long
```

```
    n=InputBox("请输入一个数")
```

```
    n=CLng(n) : r=fun(n)
```

```
    Print r
```

```
End Sub
```

程序运行后，单击命令按钮，在输入对话框中输入"345"，输出结果为\_\_\_\_\_。

6.18 在窗体上画一个命令按钮，然后编写如下事件过程：

```
Private Sub Command1_Click()
```

```
    For i=1 To 4
```

```
        x=4
```

```
        For j=1 To 3
```

```
            x=3
```

```
            For k=1 To 2
```

```
                x=x+6
```

```
            Next k
```

```
        Next j
```

```
    Next i
```

```
    Print x
```

```
End Sub
```

程序运行后，单击命令按钮，输出结果是\_\_\_\_\_。

6.19 以下程序的功能是：从键盘上输入若干个数字，当输入负数时结束输入，统计出全部数字的平均值，输出结果。请补全程序。

```
Private Sub Form_Click()
```

```
    Dim x , y As Single , z As Integer
```

```
    x=InputBox("Enter a score")
```

```
    Do while _____
```

```
        y=y+x
```

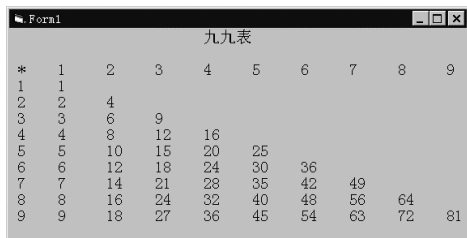
```

z=z+1
x=InputBox("Enter a score")
Loop
If z=0 Then
    z=1
End If
y= ____
Print y
End Sub

```

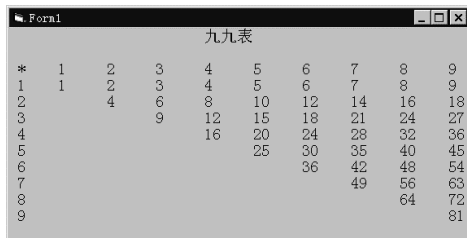
### 三、编程题

- 6.20 输入初始值，输出 100 个不能被 3 整除的数。
- 6.21 设计程序，求  $s = 1 + (1 + 2) + (1 + 2 + 3) + \cdots + (1 + 2 + 3 + \cdots + n)$  的值。
- 6.22 设  $s = 1^1 \times 2^2 \times 3^3 \times \cdots \times n^n$ ，求  $s$  不大于 400 000 时最大的  $n$  值。
- 6.23 设有一张厚为  $x$  毫米、面积足够大的纸，将它不断地对折。试问对折多少次后，其厚度可达珠穆朗玛峰的高度（8848 米）。
- 6.24 我国古代数学家张丘建在“算经”里曾提出一个世界数学史上有名的百鸡问题：“鸡翁一，值钱五；鸡母一，值钱三；鸡雏三，值钱一；百钱买百鸡，问鸡翁、母、雏各几何？”请编写程序，求出结果。
- 6.25 打印乘法“九九表”，输出结果分别如图 6-19（a）、（b）所示。



*	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

(a)



*	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9
2	2	4	6	8	10	12	14	16	18
3	3	6	9	12	15	18	21	24	27
4	4	8	12	16	20	24	28	32	36
5	5	10	15	20	25	30	35	40	45
6	6	12	18	24	30	36	42	48	54
7	7	14	21	28	35	42	49	56	63
8	8	16	24	32	40	48	56	64	72
9	9	18	27	36	45	54	63	72	81

(b)

图 6-19 “九九表”

- 6.26 所谓“水仙花数”，是指一个 3 位数，其各位数的立方和等于该数，如  $153=1^3+5^3+3^3$ ，编写程序输出所有的“水仙花数”。
- 6.27 求 1000~1100 之间的所有素数。
- 6.28 利用下述公式计算圆周率  $\pi$  的近似值：

$$\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$$

当最后一项的绝对值小于 0.000 001 时停止计算。

6.29 在窗体上输出如图 6-20 所示的图形。

6.30 马克思曾经做过这样一道趣味数学题：有 30 个人在一家小饭馆里用餐，其中有男人、女人和小孩。每个男人花了 3 先令，每个女人花了 2 先令，每个小孩花了 1 先令，一共花

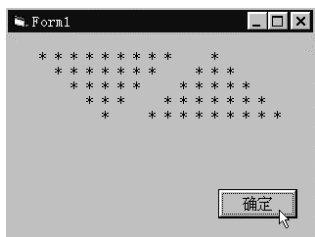


图 6-20 输出图形

去 50 先令。问男人、女人和小孩各有几人？

6.31 利用循环语句在窗体中显示不同字号大小。

6.32 用 1, 2, 3, 4 这 4 个数字组成 4 位数。编写程序，打印出所有可能的 4 位数（4 个数字可以相同），并统计出所组成的 4 位数的个数。

6.33 用“筛法”找出 1~100 之间的全部素数。

6.34 利用列表框，编写能对本学期选修课程进行课程添加、修改和删除的应用程序。

6.35 编写小学加减法算术练习程序。计算机随机给出两位数的加减法算术题，要求学生回答，答对的打“√”，答错的打“×”。将做过的题目存放在列表框中备查，并随时给出答题的正确率。

6.36 将习题 6.35 中的列表框改为组合框（下拉列表框）。

6.37 利用循环结构和列表框控件，设计“选项移动”窗体。

## 第7章 数 组

计算机处理的数据各种各样，这些数据根据有序与否可分为两类。

- 无序性数据。仅与其取值有关，而与其所在的位置无关。前面介绍的变量都是简单变量，如 a, i, x 等，并可以给简单变量赋予一个某种数据类型的数值。各个简单变量是各自独立的，与其所在的位置无关。
- 有序性数据。不仅与其取值有关，并且与其所在的位置也密切相关，例如，体育比赛的成绩就隐含着名次和成绩。

在程序设计中，利用简单变量可以解决不少问题。但是，仅使用简单变量，势必受到简单变量单独性和无序性的限制，而难以或无力解决那些不仅与取值有关，而且与其所在位置也有关的较复杂的问题。为此，需要引入功能更强的数据结构——数组。

### 7.1 数组和数组元素

数组是各种高级语言中使用广泛的程序设计方法，在讲解其应用之前，首先介绍一些有关数组和数组元素的概念。

#### 1. 数组

在程序设计中，将一组排列有序、个数有限的数作为一个整体，用一个统一的名字来表示，这些有序数据的全体称为数组。

##### (1) 数组的概念

假如有 5 个学生的成绩，这一组成绩可以用一个名字 cj 来表示，其中第 1 个学生的成绩为 80，第 2 个学生的成绩为 68，第 3 个学生的成绩为 90，第 4 个学生的成绩为 85，第 5 个学生的成绩为 95。这一组有排列顺序的数 80, 68, 90, 85, 95，就是一个数组。

在 VB 中，为了确定各数据与数组中每个元素的一一对应关系，必须给数组中的这些数据编号，即顺序号（用下标来指出顺序号，数组中也称下标变量）。

因此可以说，数组是用一个名字代表顺序排列的一组数，顺序号就是下标变量的值。简单变量是没有顺序的，无所谓谁先谁后，而数组中的各元素是有排列顺序的。

例如，在成绩数组 cj 中：

第 1 个学生的成绩用 cj(1)来表示，其值为 80；

第 2 个学生的成绩用 cj(2)来表示，其值为 68；

第 3 个学生的成绩用 cj(3)来表示，其值为 90；

第 4 个学生的成绩用 cj(4)来表示，其值为 85；

第 5 个学生的成绩用 cj(5)来表示，其值为 95。

##### (2) 数组的命名

数组的命名规则与简单变量的命名规则一样，即由 1~40 个字符组成，组成的字符可以

是字母、数字和小数点，并且必须以字母开头。例如：a, x, xscj 等。

(3) 数组的维数

数组中下标的个数称为数组的维数。

如果数组中的所有元素能按行（或列）顺序排成一行（或一列），也就是说，用一个下标便可以确定它们各自所处的位置，这样的数组称为一维数组。

如果数组中的所有元素能按行、列顺序排成一个矩阵，换句话说，必须用两个下标才能确定它们各自所处的位置，这样的数组称为二维数组。

依次类推，三个下标的数组，构成三维数组。有多少个下标的数组，就构成多少维的数组，如四维数组、五维数组等。通常又把二维以上的数组称为多维数组。

例如：

a(10)	为一维数组
x(2,3)	为二维数组
b(4,5,6)	为三维数组

2. 数组元素

在同一数组中，构成该数组的元素称为数组元素。

(1) 数组中的元素

组成数组的各个元素一般为变量，由于这些变量公用一个变量名，即它们所在的数组名，因此，必须要通过下标才能相互区别，故数组元素也称为下标变量。

在 VB 中，引用数组中的某一元素，要指出数组名和用括号括起来的数组元素在数组中的位置（顺序号）的下标，即下标变量的标识为：

〈数组名〉(〈下标表〉)

其中，〈下标表〉是指一个或者几个下标（代表一维或者几维），各下标之间应该用逗号分隔，例如：

a(5)	代表数组 a 中顺序号为 5 的那个元素
x(26)	代表数组 x 中顺序号为 26 的那个元素
c(2,3)	代表数组 c 中第 2 排第 3 列的那个元素

(2) 下标的使用说明

下标的使用方法如下。

- ① 下标放在数组名后的括号内。例如：x(10), a(4,5), b(2,3,4)等。
- ② 下标可以是常量、变量或表达式。例如：a(3), x(i), a(10-n,n+2)等。
- ③ 下标反映的是在数组中的位置。下标值若为非整数，系统将按四舍五入自动取整，其值的范围在-32 768~32 767 之间。例如：下标变量 x(3.4)，系统将按四舍五入自动取整后使用，即 x(3)。
- ④ 下标变量与简单变量一样，可以被赋值和引用。引用数组元素时的下标值应该在下限与上限之间，否则系统将显示“Subscript out range”（下标超出范围）的出错信息。

3. 数组的类型

在 VB 中，数据有多种数据类型，相应地，数组也有多种类型。可以声明任何基本数据

类型的数组，包括用户自定义类型和对象变量，但是一个数组中的所有元素应该具有相同的数据类型。

当然，数据类型为 Variant（变体）型时，各个元素能够包含不同类型的数据（对象、字符串、数值等）。

4. 数组的形式

在 VB 中，根据数组元素的个数能否变化，数组分为静态数组和动态数组两种。

- 静态数组：数组元素的个数固定不变。
- 动态数组：数组元素的个数在运行时可以改变。

7.2 静态数组

静态数组是在声明时就已经确定了数组元素个数的数组。

7.2.1 声明静态数组

声明静态数组的语法格式为：

```
Dim <数组名> ( <维数定义> ) [ As <类型> ]
```

【说明】

① <维数定义> 指定数组的维数及各维的范围：

```
[ <下标下界 1> To ] <下标上界 1> [, [ <下标下界 2> To ] <下标上界 2> ] ...
```

例如，

```
Dim a( 2 To 4 ) As Integer           ' 3 个元素，下标范围为 2~4
```

```
Dim b( 5 To 12 ) As String           ' 8 个元素，下标范围为 5~12
```

② 下标的上、下界不得超过长整型数据类型的范围（-2 147 483 648~2 147 483 647）。

③ 二维数组各维之间用逗号分隔，例如：

```
Dim a( 1 To 3 , 1 To 4 ) As Double
```

④ 可以将所有这些推广到二维以上的数组，例如：

```
Dim b( 2 , 1 To 3 , 1 To 4 )
```

7.2.2 Option Base 语句

如果不指定 <下标下界>，则数组的下界由 Option Base 语句控制。语法格式为：

```
Option Base <n>
```

【说明】

① n 只能为 0 或 1。

② 如果没有使用 Option Base 语句，则默认的下界为 0，例如：

```
Dim a(4) As Integer                 ' 5 个元素，下标范围为 0~4
```

```
Dim b(20) As Double                 ' 21 个元素，下标范围为 0~20
```

③ 如果使用 Option Base 1 语句，例如：

```
Option Base 1                       ' 默认下界为 1
```

```
Dim b(2, 1 To 3, 1 To 4)
```

则建立了一个三维数组 b，大小为  $3 \times 3 \times 4$ ，元素总数为三个维数的乘积，即 36。

### 7.2.3 数组的基本操作

在建立（声明）一个数组之后，就可以使用数组了。使用数组就是对数组元素进行各种操作，例如：赋值、表达式运算、输入或输出等。

对数组元素的操作与对简单变量的操作类似，但在引用数组元素的时候要注意以下 5 点。

① 数组声明语句不仅可以定义数组、为数组分配存储控件，而且还能对数组进行初始化，使得数值型数组的元素值初始化为 0，字符型数组的元素值初始化为空等。

② 引用数组元素的方法是在数组名后的括号中指定下标，例如：

```
t = A(5) : s = B(3,4)
```

其中，A(5)表示数组 A 中索引值为 5 的元素，B(3,4)表示二维数组 B 中行下标为 3，列下标为 4 的元素。注意与数组声明语句中下标的上界相区别。

③ 引用数组元素时，数组名、数组类型和维数必须与数组声明时一致。

④ 引用数组元素时，下标值应在数组声明时所指定的范围之内。

⑤ 在同一过程中，数组与简单变量不能同名。

### 7.2.4 数组元素的输入、输出和复制

#### 1. 数组元素的输入

数组元素可以在设计时通过赋值语句输入，或者在运行时通过 InputBox 函数输入。在元素较多的情况下，一般需要使用 For 循环语句。

**【例7-1】** 利用数组 Name()存放姓名。

考虑到要在不同的过程中使用数组，所以首先在模块的通用段声明数组：

```
Dim Name(1 To 10) As String
```

数组的赋值由窗体的 Load 事件代码完成：

```
Private Sub Form_Load()
```

```
    a(1) = "陈高阳": a(2) = "赵世杰": a(3) = "李民维": a(4) = "马英丽": a(5) = "杨广民"
```

```
    a(6) = "李灵君": a(7) = "陈吉至": a(8) = "王东明": a(9) = "姜大伟": a(10) = "吴晓林"
```

```
End Sub
```

**【例 7-2】** 随机产生 10 个两位整数，放入数组中。

考虑到要在不同的过程中使用数组，所以首先在模块的通用段声明数组：

```
Dim a(1 To 10) As Integer
```

随机整数的生成由窗体的 Load 事件代码完成：

```
Private Sub Form_Load()
```

```
    Randomize
```

```
    For i = 1 To 10
```

```
        a(i) = Int(Rnd * 90) + 10
```

Next

**End Sub**

多维数组元素的输入需要通过多重循环来实现。由于 VB 中的数组是按行存储的，因此一般把控制数组第一维的循环变量放在最外层循环中。

**【例 7-3】** 设有一个  $5 \times 5$  的方阵，其中元素是由计算机随机生成的小于 100 的整数。考虑到要在不同的过程中使用数组，所以首先在模块的通用段声明数组：

```
Dim a(5, 5) As Integer
```

方阵的生成由窗体的 Load 事件代码完成：

```
Private Sub Form_Load()
```

```
Randomize
```

```
For i = 1 To 5
```

```
For j = 1 To 5
```

```
    a(i, j) = Int(Rnd * 99) + 1
```

```
Next
```

```
Next
```

```
End Sub
```

## 2. 数组元素的输出

数组元素可以在窗体或图片框中使用 Print 方法输出，也可以在多行文本框、列表框或组合框中输出。

**【例 7-4】** 将例 7-2 中的数组在窗体中按 2 行 5 列输出。  
代码如下：

```
Private Sub Form_Activate()
```

```
Cls
```

```
Print
```

```
For i = 1 To 10
```

```
    If i Mod 5 = 0 Then
```

```
        Print a(i)
```

```
    Else
```

```
        Print a(i); "  ";
```

```
    End If
```

```
Next
```

```
End Sub
```

**【例 7-5】** 将例 7-3 中的数组在列表框中按 5 行 5 列输出。  
代码如下：

```
Private Sub Command1_Click()
```

```
List1.Clear
```

```
Dim p As String
```

```
For i = 1 To 5
```



```

p = ""
For j = 1 To 5
    p = p & Format(a(i, j), "!@@@")
Next
List1.AddItem p, i - 1
Next
End Sub

```

### 3. 数组元素的复制

单个的数组元素可以像简单变量那样从一个数组复制到另一个数组中，若要复制整个数组则仍要使用 For 循环语句。

#### 7.2.5 数组的初始化

给数组中的各个元素赋初值，称为数组的初始化。除了前面介绍的数组元素的输入方法之外，VB 还提供了 Array 函数，用于在程序中利用代码对数组进行初始化。Array 函数的语法格式为：

〈数组变量名〉 = Array(〈数组元素值〉)

##### 【说明】

① 〈数组变量名〉是已经声明的变量名称——用做数组使用，该变量必须是 Variant 型的。

② 〈数组元素值〉是准备赋给数组各元素的值列表，各值之间用逗号分开。如果不提供参数，则创建一个长度为 0 的数组。

例如，在下面的代码中，第一条语句创建一个 Variant 型变量 A，第二条语句将一个数组赋给变量 A，最后一条语句将该数组的第二个元素的值赋给另一个变量 B。

```

Dim A As Variant
A = Array(10,20,30)
B = A(2)

```

③ 使用 Array 函数创建的数组的下界一般由 Option Base 语句指定的下界来决定。

④ 没有作为数组声明的 Variant 变量也可以表示数组。除了长度固定的字符串及用户定义类型之外，Variant 变量可以表示任何类型的数组。例如，下面的语句创建了一个包含字符串的数组：

```

Dim students As Variant
students = Array("王大名", "王平", "李小双", "李丽", "张大强", "武刚")

```

注意，实际上，Array 函数返回一个包含数组的 Variant 变量。尽管一个包含数组的 Variant 变量和一个元素为 Variant 型的数组在概念上有所不同，但对数组元素的访问方式是相同的。

#### 7.2.6 静态数组使用示例

##### 【例 7-6】 Fibonacci 数列问题。

Fibonacci 数列问题起源于一个古典的有关兔子繁殖的问题。假设在第 1 个月时有一对小兔子，第 2 个月时成为大兔子，第 3 个月时成为老兔子，并生出一对小兔子（一对老，一对

小)。第 4 个月时老兔子又生出一对小兔子，上个月的小兔子变成大兔子（一对老，一对大，一对小）。第 5 个月时上个月的大兔子成为老兔子，上个月的小兔子变成大兔子，两对老兔子生出两对小兔子（两对老，一对中，两对小）……

这样，各月的兔子对数为：1, 1, 2, 3, 5, 8, …。

这就是 Fibonacci 数列。其中第  $n$  项的计算公式为：

$$\text{Fib}(n) = \text{Fib}(n-1) + \text{Fib}(n-2)$$

设计步骤如下。

（1）建立应用程序用户界面并设置对象属性，设置如图 7-1 所示。

（2）编写代码。只需编写“确定”命令按钮的 Click 事件代码：

```
Private Sub Command1_Click()  
    Dim f(15) As Integer  
    List1.Clear  
    f(1) = 1: f(2) = 1  
    p = Format("Fib(" & 1 & "):" & ", " & Format(f(1), "#####")  
    List1.AddItem p, 0  
    For i = 3 To 15  
        f(i) = f(i - 1) + f(i - 2)  
        p = Format("Fib(" & i & "):" & ", " & Format(f(i), "#####")  
        List1.AddItem p, i - 1  
    Next  
End Sub
```

【例 7-7】 随机生成 10 个互不相同的两位数，然后将这些数按由小到大的顺序显示出来，如图 7-2 所示。



图 7-1 求 Fibonacci 数列



(a)



(b)

图 7-2 排序

【分析】 这是一个排序问题，排序的方法很多，下面介绍的是比较排序法。

设有 10 个数存放在数组 A 中，分别表示为：

A(1), A(2), A(3), A(4), A(5), A(6), A(7), A(8), A(9), A(10)

第 1 轮：先将 A(1)与 A(2)比较，若  $A(2) < A(1)$ ，则将 A(1), A(2)中的值互换——A(1)中存放较小者。再将 A(1)与 A(3), …, A(10)比较，并依次做出同样的处理——10 个数中的最小者放入 A(1)中。

第 2 轮：将 A(2)与 A(3), …, A(10)比较，并依次做出同样的处理——第 1 轮余下的 9 个

数中的最小者放入 A(2)中。

继续进行第 3 轮、第 4 轮……直到第 9 轮后，余下的 A(10)自然就是 10 个数中的最大者。至此，10 个数已按从小到大顺序存放在 A(1)~A(10)中。

设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性，如图 7-2 所示。

(2) 编写代码。

考虑到要在不同的过程中使用数组，所以首先在模块的通用段声明数组：

```
Dim a(1 To 10) As Integer
```

随机整数的生成由窗体的 Load 事件代码完成：

```
Private Sub Form_Load()
```

```
    Dim p As String
```

```
    Randomize
```

```
    p = ""
```

```
    For i = 1 To 10
```

```
        Do
```

```
            x = Int(Rnd * 90) + 10
```

```
            yes = 0
```

```
            For j = 1 To i - 1
```

```
                If x = a(j) Then yes = 1: Exit For
```

```
            Next
```

```
        Loop While yes = 1
```

```
        a(i) = x
```

```
        p = p & Str(a(i)) & ", "
```

```
    Next
```

```
    Label1.Caption = LTrim(Left(p, Len(p) - 1))
```

```
    Label2.Caption = ""
```

```
End Sub
```

编写“排序”按钮 Command2 的 Click 事件代码：

```
Private Sub Command2_Click()
```

```
    For i = 1 To 9
```

```
        For j = i + 1 To 10
```

```
            If a(i) > a(j) Then
```

```
                t = a(i): a(i) = a(j): a(j) = t
```

```
            End If
```

```
        Next
```

```
    Next
```

```
    p = Str(a(1))
```

```
    For i = 2 To 10
```

```
        p = p & ", " & Str(a(i))
```

```
Next
Label12.Caption = LTrim(p)
```

End Sub

编写“重置”按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()
    Form_Load
End Sub
```

【例 7-8】 设有 10 位学生的数学、语文、外语 3 门成绩，见表 7-1。

表 7-1 学生成绩表

姓 名	数 学	语 文	外 语
陈高阳	89	85	91
赵世杰	75	78	84
李民维	64	82	72
马英丽	88	68	64
杨广民	79	79	87
李灵君	91	88	87
陈吉至	68	73	64
王东明	58	68	65
姜大伟	76	81	88
吴晓林	78	89	82

试编写程序（界面如图 7-3 所示），实现：

- （1）成绩的查询；
- （2）各科平均分数的计算；
- （3）显示各科平均分以下的学生姓名。

【分析】 本例所涉及的原始数据既有字符型又有数值型。因为在 VB 中，数组元素的类型应当一致（除非声明为 Variant 型），所以考虑用两个数组来存放原始数据：一个为字符型，用于保存姓名；另一个为数值型，用于保存各科成绩。必须注意两个数组的关系，即同一下标对应同一个人。

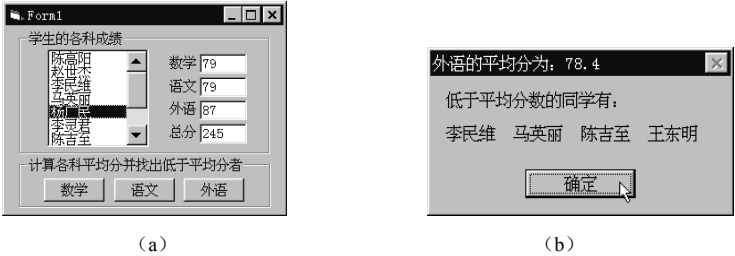


图 7-3 成绩管理

设计步骤如下。

- （1）建立应用程序用户界面并设置对象属性。

新建一个工程，进入窗体设计器，增加两个框架 Frame1 和 Frame2。激活 Frame1 后，在

其中增加一个列表框 List1, 4 个标签 Label1~Label4, 4 个文本框 Text1~Text4。激活 Frame2 后, 在其中增加 3 个命令按钮 Command1~Command3, 并修改各个控件的属性, 如图 7-3 (a) 所示。

(2) 编写代码。

考虑到要在不同的过程中使用数组, 所以首先在模块的通用段声明数组:

```
Dim a(1 To 10) As String, b(1 To 10, 1 To 4) As Integer
```

数组的赋值由窗体的 Load 事件代码完成:

```
Private Sub Form_Load()  
a(1) = "陈高阳": b(1, 1) = 89: b(1, 2) = 85: b(1, 3) = 91  
a(2) = "赵世杰": b(2, 1) = 75: b(2, 2) = 78: b(2, 3) = 84  
a(3) = "李民维": b(3, 1) = 64: b(3, 2) = 82: b(3, 3) = 72  
a(4) = "马英丽": b(4, 1) = 88: b(4, 2) = 68: b(4, 3) = 64  
a(5) = "杨广民": b(5, 1) = 79: b(5, 2) = 79: b(5, 3) = 87  
a(6) = "李灵君": b(6, 1) = 91: b(6, 2) = 88: b(6, 3) = 87  
a(7) = "陈吉至": b(7, 1) = 68: b(7, 2) = 73: b(7, 3) = 64  
a(8) = "王东明": b(8, 1) = 58: b(8, 2) = 68: b(8, 3) = 65  
a(9) = "姜大伟": b(9, 1) = 76: b(9, 2) = 81: b(9, 3) = 88  
a(10) = "吴晓林": b(10, 1) = 78: b(10, 2) = 89: b(10, 3) = 82
```

```
End Sub
```

在列表框中显示姓名由窗体的 Activate 事件代码完成:

```
Private Sub Form_Activate()  
For n = 1 To 10  
List1.AddItem a(n), n - 1  
b(n, 4) = b(n, 1) + b(n, 2) + b(n, 3)  
Next  
Text1.Text = "": Text2.Text = "": Text3.Text = "": Text4.Text = ""
```

```
End Sub
```

查阅学生的各科成绩由列表框的 Click 事件代码完成:

```
Private Sub List1_Click()  
n = List1.ListIndex + 1  
Text1.Text = b(n, 1)  
Text2.Text = b(n, 2)  
Text3.Text = b(n, 3)  
Text4.Text = b(n, 4)
```

```
End Sub
```

各科平均分数的计算及显示各科平均分以下的学生姓名的功能分别由 3 个命令按钮的 Click 事件代码分别完成:

```
Private Sub Command1_Click()  
s = 0  
For n = 1 To 10
```

```

        s = s + b(n, 1)
    Next
    s = s / 10
    p = ""
    For n = 1 To 10
        If b(n, 1) < s Then p = p & a(n) & "  "
    Next
    MsgBox "低于平均分数的同学有: " & Chr(13) & Chr(13) & p, 0, "数学的平均分为: " & s
End Sub

Private Sub Command2_Click()
    s = 0
    For n = 1 To 10
        s = s + b(n, 2)
    Next
    s = s / 10
    p = ""
    For n = 1 To 10
        If b(n, 2) < s Then p = p & a(n) & "  "
    Next
    MsgBox "低于平均分数的同学有: " & Chr(13) & Chr(13) & p, 0, "语文的平均分为: " & s
End Sub

Private Sub Command3_Click()
    s = 0
    For n = 1 To 10
        s = s + b(n, 3)
    Next
    s = s / 10
    p = ""
    For n = 1 To 10
        If b(n, 3) < s Then p = p & a(n) & "  "
    Next
    MsgBox "低于平均分数的同学有: " & Chr(13) & Chr(13) & p, 0, "外语的平均分为: " & s
End Sub

```

**【例 7-9】** 将下列字符存放到数组中，并以倒序打印出来：

a b q r s t w x y e m n

**【分析】**把这 12 个字符存放到数组 a(12)中，首先依次读取，然后利用 For 循环，设步长为-1，初值为 12，终值为 1，实现倒序输出。

设计步骤如下。

(1) 建立用户界面并设置对象属性，如图 7-4 所示。

(2) 编写事件代码。

编写“显示倒序”命令按钮 Command1 的 Click 事件代码为：

**Private Sub Command1\_Click()**

```

Dim x As Integer, a(1 To 12) As String
a(1) = "a" : a(2) = "b" : a(3) = "q" : a(4) = "r" : a(5) = "s" : a(6) = "t"
a(7) = "w" : a(8) = "x" : a(9) = "y" : a(10) = "e" : a(11) = "m" : a(12) = "n"
For x = 1 To 12
    Label1.Caption = Label1.Caption & a(x)      ' 按原字符顺序输出
Next x
For x = 12 To 1 Step -1
    Label2.Caption = Label2.Caption & a(x)      ' 按倒序输出
Next x

```

**End Sub**

运行程序，结果如图 7-4 所示。

**【例 7-10】** 建立并输出一个  $10 \times 10$  的矩阵，使该矩阵两条对角线上的元素均为 1，其余元素均为 0，如图 7-5 所示。

**【分析】** 由于矩阵由行、列组成，需要双下标才能确定某一元素的位置，所以，使用二维数组来表示该矩阵。设行用  $n$  表示，列用  $m$  表示，则主对角线元素即为行与列相等的元素，即  $n = m$ ，而次对角线元素的行列下标则满足： $n = 11 - m$ 。

在窗体中使用图片框控件 Picture1，如图 7-5 所示，当然也可以用列表框控件来显示矩阵的元素。

编写窗体 **Form1** 的 Activate 事件代码：

**Private Sub Form\_Activate()**

```

Dim s(10, 10) As Integer
For n = 1 To 10                                ' 控制行数
    For m = 1 To 10                            ' 控制列数
        If n = m Or n = 11 - m Then           ' 对角线元素
            s(n, m) = 1
        Else                                   ' 非对角线元素
            s(n, m) = 0
        End If
    Next
Next
Picture1.Print
For n = 1 To 10                                ' 控制行数
    For m = 1 To 10                            ' 控制列数
        Picture1.Print Tab(m * 3); s(n, m);   ' 输出各元素的值
    Next
    Print                                       ' 换行
Next

```

**End Sub**

运行程序，结果如图 7-5 所示。



图 7-4 倒序输出字符

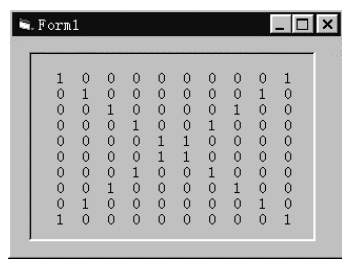


图 7-5 输出对角线元素为 1 的矩阵

### 7.3 动态数组

在数组的使用中，有时在程序设计阶段并不知道数组所需大小，而无法声明正确的数组大小；或者在某个过程中需要一个特别大的数组，如果在程序一开始处就声明一个大数组，则保存该数组的内存长期被占用，会降低系统效率。遇到这些情况，都可以使用动态数组，对于前一种情况，可以一边输入数据一边随着数据量的增加而重新声明数组的大小；而对于后一种情况，可在需要使用特别大数组的过程中重新声明数组大小，离开该过程前取消该数组。

#### 7.3.1 创建动态数组

创建动态数组的步骤如下。

(1) 用 **Public** 语句（公用数组）、**Dim** 语句（模块级数组）、**Static** 语句或 **Private** 语句（局部数组），在过程中声明一个未指明大小及维数的空数组，这样就将数组声明为动态数组了。语法格式为：

**Public | Private | Dim | Static** <数组名> () **As** <类型>

(2) 用 **ReDim** 语句分配实际的元素个数，语法格式为：

**ReDim** [**Preserve**] <数组名> ( <维数定义> ) [**As** <类型> ]

例如，第一次声明在模块级所建立的动态数组 a：

**Dim** a() **As** Integer

然后，在过程中给数组分配空间：

**Private Sub** Form\_Activate()  
.....

**ReDim** a(9, 19)

**End Sub**

这里的 **ReDim** 语句给 a 分配一个 10×20 的整数矩阵（元素总数为 200）空间。

#### 【说明】

① **ReDim** 语句只能出现在过程中。与 **Dim** 语句、**Static** 语句不同，**ReDim** 语句是一个可执行语句。由于这一语句的存在，应用程序在运行时将执行一个操作。

② 每次运行程序时，**ReDim** 语句都会清除数组内容，当前存储在数组中的值将全部丢失，VB 重新将数组元素的值置为 0（对 **Numeric** 数组），或置为零长度字符串（对 **String** 数



组)或置为 Empty (对 Variant 数组)。此时可以用 Preserve 关键字保留原先的数据。

③ 声明动态数组的时候并不指定数组的维数,数组的维数由第一次出现的 ReDim 语句指定。

④ 对于任一维数,ReDim 语句都能改变元素的数目及其上下界,但是,数组的维数不能改变。

### 7.3.2 保留动态数组的原有数据

使用 ReDim 语句时,将清除数组中的原有数据。但是,有时需要改变数组大小而又不丢失数组中的数据,这时就可以使用具有 Preserve 关键字的 ReDim 语句。

#### 1. 具有 Preserve 关键字的 ReDim 语句

使用具有 Preserve 关键字的 ReDim 语句来增加数组大小,又不丢失原数据,代码如下:

```
ReDim a(2,4)
.....
ReDim Preserve a(2,6)
```

则原数组中数据均可保留,且增加了 a(1,5), a(1,6), a(2,5), a(2,6)这 4 个位置。

如果声明 ReDim a(3,4),将会清除原数组内容。

使用 Preserve 关键字,只能改变多维数组中最后一维的上界,而不能改变维数。如果数组就是一维的,则可以重定义该维的大小,因为它是最末维,也是仅有的一维;如果数组是二维或多维的,则只能改变其最末维才可同时保留数组中的内容。如果改变了其他维或最后一维的下界,运行时就会出错。

#### 2. 动态数组使用示例

**【例 7-11】** 编写程序,输出杨辉三角形(国外称 Pascal 三角形)。杨辉三角形每行的第一列和最后一列均为 1,其余各项的值都是其上一行中前一列元素与后一列元素之和,如图 7-6 所示。

**【分析】** 杨辉三角形中的各行是二项式 $(a + b)^n$ 展开式中各项的系数。由图 7-8 的排列格式可以看出,上一行同一列中没有元素时则认为是 0。由此可得算法为:

$$A(i, j) = A(i - 1, j - 1) + A(i - 1, j)$$

设计步骤如下。

(1) 建立用户界面并设置对象属性,如图 7-7 所示。



图 7-6 杨辉三角形

图 7-7 建立用户界面并设置对象属性

(2) 编写事件代码。

在模块的通用段声明数组为一个动态数组：

```
Dim a()
```

为了能输入行数并在按 Enter 键后可以得到各项，编写文本框 Text1 的 KeyPress 事件代码如下：

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
```

```
Dim n As Integer
```

```
If KeyAscii = 13 Then ' 按 Enter 键后执行
```

```
    n = Val(Text1.Text) ' 在文本框中输入的行数
```

```
    If n > 10 Then ' 设定不超过 10 行
```

```
        MsgBox "请不要超过 10!" ' 消息框
```

```
        Exit Sub ' 退出过程
```

```
    End If
```

```
    ReDim a(n, n) ' 分配动态数组实际的元素个数
```

```
    For i = 1 To n
```

```
        a(i, 1) = 1: a(i, i) = 1 ' 使得每行两边的元素值为 1
```

```
    Next
```

```
    Print Tab(20); Format(1, "!@@@") & Chr(13)
```

```
    Print Tab(18); Format(1, "!@@@") & Space(2) & Format(1, "!@@@") & Chr(13)
```

```
    For i = 3 To n
```

```
        Print Tab(20 - i * 2); Format(a(i, 1), "!@@@") & Space(2);
```

```
        For j = 2 To i - 1
```

```
            a(i, j) = a(i - 1, j - 1) + a(i - 1, j)
```

```
            Print Format(a(i, j), "!@@@@@");
```

```
        Next
```

```
        Print Space(2) & Format(a(i, i), "!@@@@") & Chr(13)
```

```
    Next
```

```
End If
```

```
End Sub
```

运行程序，在文本框中输入行数，按 Enter 键后，显示出杨辉三角形，如图 7-7 所示。

## 7.4 For Each…Next 语句

与 For…Next 循环语句类似，For Each…Next 循环语句也是用来执行指定重复次数的。但是，For Each…Next 语句专门用于数组或对象集中的每个元素。

### 1. For Each…Next 语句的语法格式

For Each…Next 语句的语法格式为：

```
For Each 〈成员〉 In 〈数组〉
```

[ 〈语句组〉 ]

[ Exit For ]

Next [ 〈成员〉 ]

#### 【说明】

- ① 〈成员〉是一个 Variant（变体）型变量，代表数组中的每个元素。
- ② 〈数组〉是一个数组名，没有括号和上下界。

## 2. For Each...Next 语句的执行过程

用 For Each...Next 语句可以对数组中的元素进行处理，包括查询、显示或读取。它所重复执行的次数由数组中元素的个数确定，也就是说，数组中有多少个元素，就自动重复执行多少次。例如：

```
Dim a(1 To 8)
For Each x In a
    Print x;
Next x
```

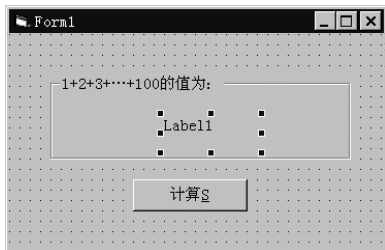
上面程序中的 Print 语句重复 8 次（因为数组 a 中有 8 个元素），每次输出数组中一个元素的值。这里的 x 类似于 For...Next 循环中的循环控制变量，但不需要为其提供初值和终值，而是根据数组元素的个数确定执行循环体的次数。此外，x 的值处于不断的变化之中：开始执行时，x 是数组第一个元素的值；执行完一次循环体后，x 变为数组第二个元素的值；……；当 x 为最后一个元素的值时，执行最后一次循环。

在数组操作中，使用 For Each...Next 语句比 For...Next 语句更方便，因为它不需要指明循环的条件。

## 3. For Each...Next 语句使用示例

**【例 7-12】** 用 For Each...Next 语句，求  $1+2+3+\cdots+100$  的值。  
设计步骤如下。

- （1）建立用户界面并设置对象属性，如图 7-8（a）所示。



(a)



(b)

图 7-8 建立用户界面并设置对象属性

- （2）编写事件代码。

编写“计算”命令按钮 Command1 的 Click 事件代码：

```

Private Sub Command1_Click()
    Dim x(100), a                ' 声明数组和变量
    For i = 1 To 100              ' 为数组元素赋值
        x(i) = i
    Next i
    For Each a In x()              ' 求和
        s = s + a
    Next
    Label1.Caption = s            ' 输出结果
End Sub

```

运行程序，结果如图 7-8（b）所示。

【例 7-13】 求例 7-2 随机整数中的最大值、最小值和平均值。  
编写命令按钮 Command1 的 Click 事件代码：

```

Private Sub Command1_Click()
    Dim n As Integer, m As Integer, s As Single
    m = 100: n = 0: s = 0
    For Each x In a
        If x > n Then n = x
        If x < m Then m = x
        s = s + x
    Next
    MsgBox "最大值为" & n & Chr(13) & "最小值为" & m & Chr(13) & "平均值为" & s / 10
End Sub

```

【说明】不能使用 For Each…Next 语句对普通的数组元素进行赋值操作，因为语句中的〈成员〉表示数组元素的值，而不表示数组元素本身，但是可以对控件数组中每个控件的属性进行赋值操作。

## 7.5 控件数组

许多同样的数据类型，保存在一个变量里称为数组；同理，许多相同的控件集合在一起，就是控件数组。

### 7.5.1 控件数组的概念

控件数组是由一组相同类型的控件组成的，它们公用一个控件名，具有相同的属性。当建立控件数组时，系统给每个元素赋一个唯一的索引号（Index），通过属性窗口的 Index 属性，可以知道该控件的下标是多少。第一个控件的下标是 0。例如，控件数组 cmdName(5) 表示控件数组名为 cmdName 的第 6 个元素。

控件数组的特点为：

- 具有相同的名称（Name）；
- 用下标索引值（Index）来识别各个控件。

控件数组适用于若干个控件执行操作相似场合，控件数组共享同样的事件过程。例如，控件数组 `cmdName` 有 6 个命令按钮，则不管单击哪个命令按钮，都会调用同一个事件过程。

为了区分控件数组中的各个元素，VB 将把下标索引值传送给过程。在程序运行中，可以利用返回的索引值来识别事件是由哪个控件所引发的。

## 7.5.2 控件数组的建立

可以使用下述三种方法之一建立控件数组：

- 为控件起相同的名字；
- 复制现有控件；
- 指定控件的索引值。

### 1. 为控件起相同的名字

可以改变已有控件的名字，将一组控件建立为控件数组，具体步骤如下。

(1) 画出控件数组中要添加的控件（必须为同一类型的控件），并且决定哪一个控件作为数组中的第一个元素。

(2) 选定控件，将其 `Name` 属性设置成数组名称。

(3) 在为数组中的其他控件输入相同名称时，VB 将显示一个对话框，要求确认是否要创建控件数组。此时回答“是”，确认此操作。

例如，若控件数组第一个元素名为 `Command1`，此时选择另一个命令按钮添加到数组中，并将其名称也设置为 `Command1`，将显示这样一段信息：“已经有一个控件为'Command1'。创建一个控件数组吗？”。单击“是”按钮，确认操作，如图 7-9 所示。






图 7-9 确认创建控件数组

用这种方法添加的控件仅共享 `Name` 属性和控件类型，其他属性与最初画出控件时的值相同。

### 2. 复制现有控件

用复制、粘贴的方法建立控件数组，具体步骤如下。

- (1) 画出控件数组中的第一个控件。
- (2) 当控件获得焦点时，单击“复制”按钮 。
- (3) 单击“粘贴”按钮 ，VB 将显示一个对话框询问是否确认创建控件数组。回答“是”，确认操作，将得到控件数组中的第二个控件。
- (4) 继续单击“粘贴”按钮 ，可得到控件数组中的其他控件。

每个新数组元素的索引值与其添加到控件数组中的次序相同，如图7-10中第二次粘贴的 Option1 控件，其 Index 的值为 2。并且在添加控件时，大多数可视属性，如高度、宽度和颜色等，都将从数组中第一个控件复制到新控件中。

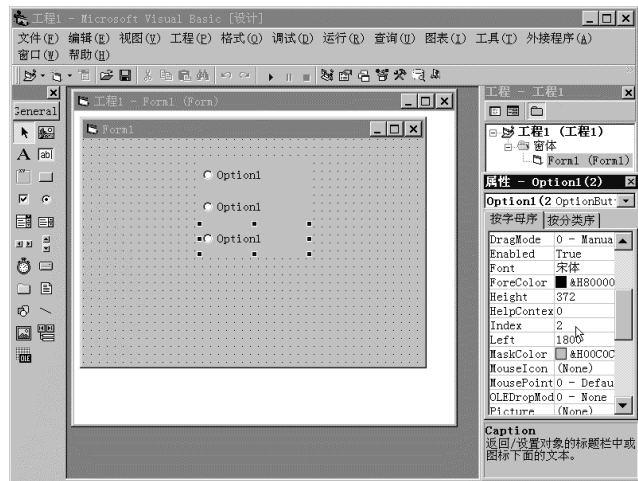


图 7-10 新数组元素的索引值与其添加到控件数组中的次序一样

### 3. 指定控件的索引值

可以直接指定控件数组的索引值，来建立控件数组。其具体步骤如下。

- (1) 绘制控件数组中的第一个控件。
- (2) 将其 Index 属性索引值改为 0。
- (3) 复制控件数组中的其他控件。

这时，将不会出现对话框询问是否确认创建控件数组。

## 7.5.3 控件数组使用示例

控件数组可以在命令按钮组、单选钮组、复选框组、文本框组、标签组等控件中使用。

### 1. 命令按钮组

**【例 7-14】** 在例 7-8 中使用命令按钮控件数组。

**【分析】** 例 7-8 中计算平均分数的 3 个命令按钮，其 Click 事件代码非常相似，使用命令按钮组可以使代码更加简捷。

在例 7-8 的基础上加以修改，具体步骤如下。

- (1) 修改命令按钮的属性设置。

首先将 Command1 的 Index 属性改为 0，然后依次将 Command2 与 Command3 的 Name

属性改为 Command1，此时，两个按钮的 Index 属性自动改为 1, 2。这样，3 个命令按钮变为一个命令按钮控件数组 Command1。

(2) 修改 Command1 的 Click 事件代码。

首先在原有 Command1 的 Click 事件过程中增加参数声明：

Index As Integer

然后修改代码为：

**Private Sub Command1\_Click(Index As Integer)**

k = Index

Select Case k

Case 0

t = "数学的平均分为："

Case 1

t = "语文的平均分为："

Case 2

t = "外语的平均分为："

End Select

s = 0

For n = 1 To 10

s = s + b(n, k + 1)

Next

s = s / 10

p = ""

For n = 1 To 10

If b(n, k + 1) < s Then p = p & a(n) & " "

Next

MsgBox "低于平均分数同学有：" & Chr(13) & Chr(13) & p, 0, t & s

**End Sub**

最后，删除原来 Command2 与 Command3 的事件过程代码。

## 2. 单选按钮

**【例 7-15】** 如图 7-11 所示，使用控件数组选择文本字体。

用控件数组可以使代码更为简捷，操作步骤如下。

(1) 在窗体上添加一个文本框、一个标签和 4 个单选按钮控件，并将这 4 个单选按钮控件改为单选按钮控件数组 Option1(0)~Option1(3)。属性设置参见图 7-11。

(2) 编写单选按钮控件数组 Option1() 的 Click 事件代码：

**Private Sub Option1\_Click(Index As Integer)**

Select Case Index

Case 0

Text1.FontName = "宋体"

```
Case 1
    Text1.FontName = "隶书"
Case 2
    Text1.FontName = "黑体"
Case 3
    Text1.FontName = "楷体_GB2312"
End Select
End Sub
```

3. 复选框组

【例 7-16】 如图 7-12 所示，使用复选框组控制文本字体风格。



图 7-11 使用单选按钮组

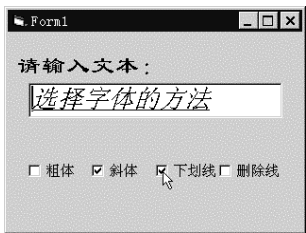


图 7-12 使用复选框组

操作步骤如下。

(1) 在例 7-15 的基础上修改界面，添加 4 个复选框控件，并将 4 个复选框控件改为复选框组 Check1(0)~Check1(3)。属性设置如图 7-12 所示。

(2) 编写复选框组的 Click 事件代码：

```
Private Sub Check1_Click(Index As Integer)
    Select Case Index
        Case 0
            Text1.FontBold = Check1(0).Value
        Case 1
            Text1.FontItalic = Check1(1).Value
        Case 2
            Text1.FontUnderline = Check1(2).Value
        Case 3
            Text1.FontStrikethru = Check1(3).Value
    End Select
    Text1.SetFocus
End Sub
```

4. 文本框组

【例 7-17】 设计程序，演示例 7-7 的比较排序过程，使用控件数组可以使排序过程更加生动，如图 7-13 所示。





图 7-13 比较法排序

设计步骤如下。

(1) 建立应用程序用户界面。新建一个工程，进入窗体设计器。首先增加一个用做容器的框架控件 Frame1，选中 Frame1，在其中增加一个文本框控件数组 Text1(1)~Text1(10)，一个标签控件数组 Label1(0)~Label1(9)和一个标签 Label2。然后在窗体中增加一个命令按钮控件数组 Command1(0)~Command1(1)。如图 7-14 (a) 所示。将 Frame1 控件的 Height 属性改小，再在窗体中增加一个标签 Label3，如图 7-14 (b) 所示。

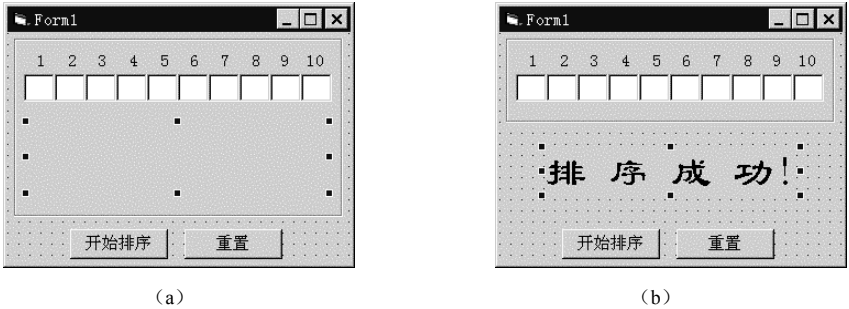


图 7-14 建立应用程序界面

(2) 设置对象属性，参见表 7-2。

表 7-2 对象属性设置

对 象	属 性	属 性 值
Command1(0)	Caption	开始
Command1(1)	Caption	重置
Frame1	Caption	
Label1(0)~Label1(9)	Caption	依次改为: 1, 2, ..., 10
Label2	Caption	
Label3	Caption	排序成功
Text1(1)~Text1(10)	Alignment	2 - Center
	Text	(无)

(3) 设计代码。

在窗体的通用过程中声明数组变量：

Dim a(10) As TextBox ' 显示数据的对象数组

编写窗体的 Activate 事件代码：

**Private Sub Form\_Activate()**

```

Randomize
For i = 1 To 10
    Set a(i) = Text1(i)
    a(i).Text = Int(Rnd * 199) - 99
    a(i).BackColor = RGB(255, 255, 255)
Next
Frame1.Height = 2256

```

**End Sub**

编写命令按钮控件数组 Command1()的 Click 事件代码:

**Private Sub Command1\_Click(Index As Integer)**

```

Select Case Index
    Case 0
        Command1(0).Enabled = False
        For i = 1 To 9
            a(i).BackColor = RGB(255, 0, 255)
            Label2.Caption = " 以 A(" & Trim(i) & ")为擂主与后边的元素依次比较,如有比 A(" & _
                Trim(i) + ")小者, 则与 A(" & Trim(i) & ")交换。"
            For j = i + 1 To 10
                a(j).BackColor = RGB(255, 0, 255)
                MsgBox "比较 A(" & Trim(i) & ")和 A(" & Trim(j) & ")", , "比较法排序"
                If Val(a(i).Text) > Val(a(j).Text) Then
                    p1 = "交换 A(" & Trim(i) & ")和 A(" & Trim(j) & ")"
                    p2 = "A(" & Trim(i) & ") > A(" & Trim(j) & ")"
                    MsgBox p1, , p2
                    t = a(i).Text: a(i).Text = a(j).Text: a(j).Text = t
                End If
                a(j).BackColor = RGB(255, 255, 255)
            Next j
            a(i).BackColor = RGB(0, 255, 0)
        Next i
        a(10).BackColor = RGB(0, 255, 0)
        Label2.Caption = ""
        Frame1.Height = 1000
    Case 1
        Form_Activate
        Command1(0).Enabled = True
End Select

```

**End Sub**

## 习题 7

### 一、选择题

7.1 设有声明语句如下，则数组 b 中全部元素的个数为 ( )。

Dim b(-1 To 10,2 To 9,20) As Integer

A) 2310                      B) 2016                      C) 1500                      D) 1658

7.2 设有数组定义语句 Dim a(5) As Integer, 下列给数组元素赋值的语句错误的是 ( )。

A) a(3)=3                      B) a(3)=InputBox("input data")

C) a(3)=List1.ListIndex                      D) a=Array(1,2,3,4,5,6)

7.3 在窗体上画一个名称为 Label1 的标签，然后编写如下事件过程：

**Private Sub Form\_Click()**

Dim arr(10,10) As Integer

Dim i As Integer, j As Integer

For i=2 To 4

For j=2 To 4

arr(i,j)=i\*j

Next j

Next i

Label1.Caption=Str(arr(2,2)+arr(3,3))

**End Sub**

程序运行后，单击窗体，在标签中显示的内容是 ( )。

A) 12                      B) 13                      C) 14                      D) 15

7.4 阅读程序：

Option Base 1

Dim arr() As Integer

**Private Sub Form\_Click()**

Dim i As Integer, j As Integer

ReDim arr(3,2)

For i=1 To 3

For j=1 To 2

arr(i,j)=i\*2+j

Next j

Next i

ReDim Preserve arr(3,4)

For j=3 To 4

arr(3,j)=j+9

Next j

Print arr(3,2)+arr(3,4)

**End Sub**

程序运行后，单击窗体，输出结果为（ ）。

- A) 21                      B) 13                      C) 8                      D) 25

7.5 在窗体上画一个名称为 Command1 的命令按钮，然后编写如下程序：

Option Base 1

**Private Sub Command1\_Click()**

Dim c As Integer, d As Integer

d=0

c=6

x=Array(2,4,6,8,10,12)

For i=1 To 6

    If x(i)>c Then

        d=d+x(i)

        c=x(i)

    Else

        d=d-c

    End If

Next i

Print d

**End Sub**

程序运行后，如果单击命令按钮，则在窗体上输出的内容为（ ）。

- A) 10                      B) 16                      C) 12                      D) 20

7.6 以下程序段的输出结果为（ ）。

Dim I, a(10), p(3)

k=5

For i=0 To 10

    a(i)=i

Next i

For i=0 To 2

    p(i)=a(i\*(i+1))

Next i

For i=0 To 2

    k=k + p(i) \* 2

Next i

Print k

- A) 20                      B) 21                      C) 56                      D) 32

7.7 假定建立了一个名为 Command1 的命令按钮数组，则以下说法错误的是（ ）。

- A) 数组中每个命令按钮的名称（Name 属性）均为 Command1  
B) 数组中每个命令按钮的标题（Caption 属性）都一样  
C) 数组中所有命令按钮可以使用同一个事件过程  
D) 用名称 Command1（下标）可以访问数组中的每个命令按钮

7.8 下列叙述中，正确的是（ ）。

- A) 控件数组的每一个成员的 **Caption** 属性值都必须相同
- B) 控件数组的每一个成员的 **Index** 属性值都必须不相同
- C) 控件数组的每一个成员都执行不同的事件过程
- D) 对已经建立的多个类型相同的控件，这些控件不能组成控件数组

7.9 在窗体上画一个名称为 **Text1** 的文本框和一个名称为 **Command1** 的命令按钮，然后编写如下事件过程：

```
Private Sub Command1_Click()  
    Dim array1(10,10)As Integer  
    Dim i As Integer,j As Integer  
    For i=1 To 3  
        For j=2 To 4  
            Array1(i,j)=i+j  
        Next j  
    Next i  
    Text1.Text=array1(2,3)+array1(3,4)
```

**End Sub**

程序运行后，单击命令按钮，在文本框中显示的值是（ ）。

- A) 15
- B) 14
- C) 13
- D) 12

## 二、填空题

7.10 用 **Dim(1, 3 to 7,10)**声明的是一个\_\_\_\_维数组。

7.11 在窗体上画一个命令按钮，然后编写如下事件过程：

**Option Base 1**

```
Private Sub Command1_Click()
```

```
    Dim a  
    A=Array(1 , 2 , 3 , 4)  
    j=1  
    For i=4 To 1 Step-1  
        s=s+a(i)*j  
        j=j*10  
    Next i  
    Print s
```

**End Sub**

运行程序，单击命令按钮，其输出结果是\_\_\_\_\_。

## 三、编程题

7.12 输入一串字符，统计各字母出现的次数，不区分大小写。

7.13 随机产生 10 个两位整数，找出其中的最大值、最小值和平均值。

7.14 利用随机函数模拟投币结果。设共投币 100 次，求“两个正面”、“两个反面”、“一正一反” 3 种情况各出现多少次。

7.15 某数组有 20 个元素，元素的值由键盘输入，要求将前 10 个元素与后 10 个元素对换，即第 1 个元素与第 20 个元素对换，第 2 个元素与第 19 个元素对换，……，第 10 个元素与第 11 个元素对换。输出数组原来各元素的值和对换后各元素的值。

7.16 有一个 6×6 的矩阵，各元素的值由键盘输入，求全部元素的平均值，并输出高于平均值的元素及它们的行、列号。

7.17 编写应用程序，输入两个数，根据不同运算符计算结果。

7.18 编写矩阵转置程序，将一个 6 行 4 列的矩阵行、列互换（其中的矩阵元素随机产生），即

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \\ a_{51} & a_{52} & a_{53} & a_{54} \\ a_{61} & a_{62} & a_{63} & a_{64} \end{bmatrix} \xrightarrow{\text{转置}} \begin{bmatrix} a_{11} & a_{21} & a_{31} & a_{41} & a_{51} & a_{61} \\ a_{12} & a_{22} & a_{32} & a_{42} & a_{52} & a_{62} \\ a_{13} & a_{23} & a_{33} & a_{43} & a_{53} & a_{63} \\ a_{14} & a_{24} & a_{34} & a_{44} & a_{54} & a_{64} \end{bmatrix}$$

7.19 编写矩阵的加法运算程序。两个相同阶数的矩阵 *A* 和 *B*（随机产生）相加，就是将相应位置上的元素相加后放到同阶矩阵 *C* 的相应位置。

7.20 输出幻方阵。幻方阵也称魔方阵，是指由自然数 1~*n*<sup>2</sup>（*n* 为奇数）构成的方阵，其各行、各列及对角线元素之和均相等。如图 7-15 所示。

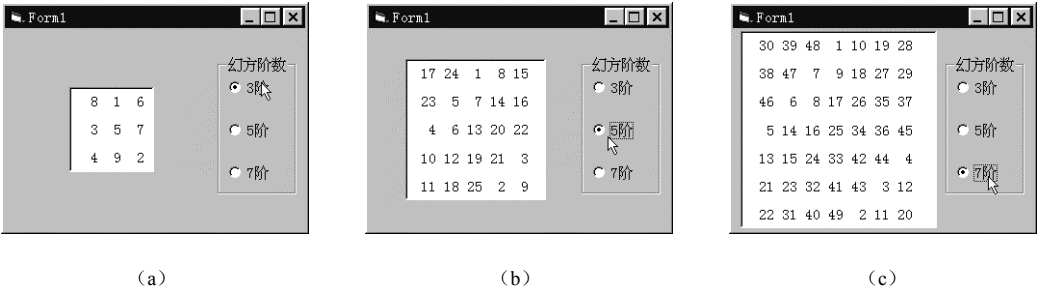


图 7-15 幻方阵

7.21 利用一维数组统计一个班学生 0~9 分，10~19 分，20~29 分，……，90~99 分及 100 分各分数段的人数。

7.22 设计一个“通信录”程序，如图 7-16（a）所示。当用户在“选择姓名”下拉列表框中选择某一人名后，在“电话号码”文本框中显示出对应的电话号码，如图 7-16（b）所示。当用户选择或取消“单位”和“住址”复选框后，将打开或关闭“工作单位”和“家庭住址”文本框，如图 7-16（c）所示。

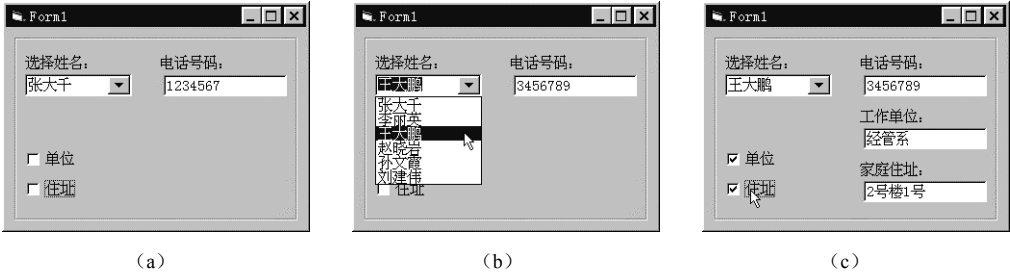


图 7-16 “通信录”程序

7.23 设计简易计算器，如图 7-17 所示。

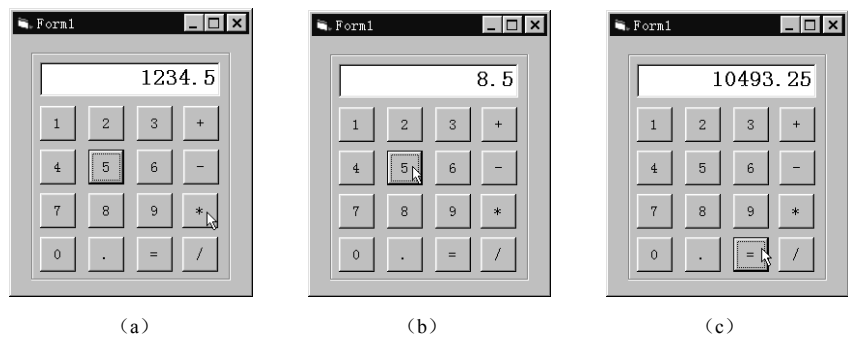


图 7-17 简易计算器

7.24 设有一个  $5 \times 5$  的方阵，其中元素是由计算机随机生成的小于 100 的整数。求：

- (1) 对角线上元素之和；
- (2) 对角线上元素之积；
- (3) 方阵中最大的元素。

## 第 8 章 过 程

使用“过程”是实现结构化程序设计思想的重要方法。结构化程序设计思想的要点之一就是对于一个复杂的问题采用模块化方法，即把一个较大的程序划分为若干个模块，每个模块只完成一个或若干个功能。这些模块通过执行一系列的语句来完成一个特定的操作过程，因此被称为“过程”。用过程编程的好处如下：

- 过程可使程序分解成离散的逻辑单元，每个单元都比无过程的整个程序更容易调试；
- 一个程序中的过程，往往不必修改或只需稍做改动，便可以被另一个程序所使用。

VB 中主要有 3 种过程。

### （1）事件过程

对象事件在用户或系统发出动作时被触发，事件响应时执行的过程就是事件过程。事件过程一般由 VB 创建，用户不能增加或删除。

### （2）通用过程

通用过程是指必须由其他过程显式调用的代码块。通用过程由用户自己创建，可以用来将复杂的程序划分为小的代码模块。在一个工程中，通用过程可以被其他过程调用，这样会提高代码的使用效率。

通用过程根据过程是否返回值，分为子（Sub）过程和函数（Function）过程两种。

- 子过程：不返回值，可以作为独立的基本语句调用。
- 函数过程：返回一个值。

### （3）属性过程

属性过程可以用来返回和设置属性的值，还可设置对象的引用。

## 8.1 事件过程

当用户对某个对象发出一个动作时，Windows 会通知 VB 产生了一个事件，VB 会自动地调用与该事件相关的事件过程，即当对象对一个事件的发生做出认定时，VB 便自动用相应事件的名字调用该事件的过程。由于名字在对象和代码之间建立了联系，所以说，事件过程是依附于窗体和控件上的。

控件事件过程的语法格式为：

```
Private Sub <控件名>_<事件名> ([ 形参表 ]  
    [ <语句组> ]  
End Sub
```

窗体事件过程的语法格式为：

```
Private Sub Form_<事件名> ([ 形参表 ]  
    [ <语句组> ]  
End Sub
```



【说明】

① 虽然用户可以手工输入首行的事件过程名，但使用模板会更方便，模板会自动将正确的过程名包括进来。

使用模板创建事件过程的方法是：在对象列表框中选定活动窗体中的对象名（如 Command1），在过程列表框中选择事件名（如 Click 事件），系统就会在代码编辑器窗口中生成该对象所选事件的过程模板，如图 8-1 所示。然后在 Sub 和 End Sub 语句之间输入代码。

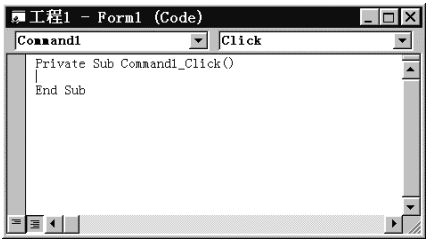


图 8-1 使用模板创建事件过程

② 事件过程名是由 VB 自动给出的，如 Command1\_Click。因此，在新控件或对象编写事件代码之前，应先设置它的 Name 属性，图 8-2 中将 Command1 的 Name 属性设置为 Cmdopen，则事件过程名就自动给出为 Cmdopen\_Click。

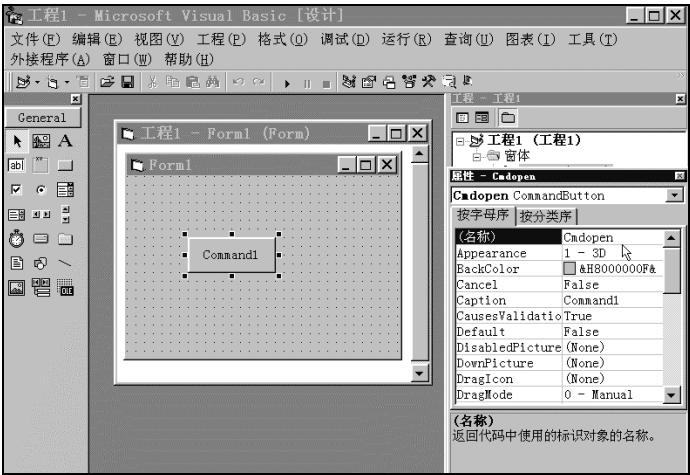


图 8-2 设置对象的 Name 属性

如果编写代码后再改变控件或对象的 Name 属性，也必须同时更改事件过程的名字。否则，控件或对象会失去与代码的联系，这时将会把它当做一个通用过程。

8.2 子过程

当有几个不同的事件过程需要执行相同的操作时，为了简化程序，可以将公共语句放入分离开的子过程（通用过程）中，并由事件过程来调用它。这样不必重复编写代码，维护程序也较容易。

子过程不与任何特定的事件相联系，只能由别的过程来调用，它可以存储在窗体或标准模块中。

### 8.2.1 创建子过程

子过程与事件过程不同，子过程不是由对象的某种事件激活的，也不依附于某一对象，故其创建的方法略有区别。建立子过程有两种方法：

- 使用“添加过程”对话框；
- 直接在代码编辑窗口中输入过程代码。

#### 1. 使用“添加过程”对话框

使用“添加过程”对话框，创建过程的步骤如下。

- (1) 打开代码编辑窗口。
- (2) 执行“工具”→“添加过程”菜单命令，打开“添加过程”对话框，如图 8-3 所示。
- (3) 在“名称”文本框中输入过程名，如“shuru”；在“类型”组中选择“子程序”项；在“范围”组中选择范围，相当于使用 **Public**（公有的）或 **Private**（私有的）关键字。
- (4) 单击“确定”按钮。可以看到在对象列表框中为“(通用)”，在过程列表框中为“shuru”，如图 8-4 所示。



图 8-3 “添加过程”对话框

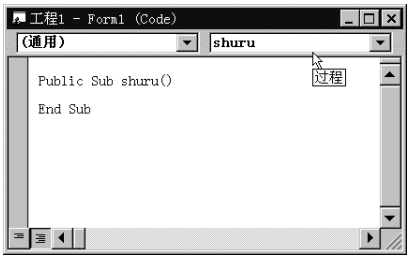


图 8-4 代码编辑窗口

#### 2. 在代码编辑窗口中输入

在代码编辑窗口中，把光标定位在已有过程的外面，然后按如下格式输入子过程：

```
[ Private | Public | Static ] Sub <过程名> ([ <形参表> )  
[ <语句组> ]  
[ Exit Sub ]  
[ <语句组> ]  
End Sub  
【说明】
```

- ① VB 默认所有模块中的子过程是公有的（**Public**），表示在应用程序中随处都可调用它们；如果选用 **Private**，则只有该过程所在模块中的程序才能调用该过程。
- ② 如果使用 **Static**（静态的）关键字，则该过程中的所有局部变量的存储空间只分配一次，且这些变量的值在整个程序运行期间都存在；如果省略 **Static**，过程每次被调用时都重新为其变量分配存储空间，当该过程结束时释放其变量的存储空间。
- ③ <过程名> 与变量名的命名规则相同，长度不得超过 40 个字符。
- ④ 在过程内，不能再定义过程，但可以调用其他子过程或函数过程。

### 8.2.2 调用子过程

调用子过程的方法有两种。

使用 Call 语句，语法格式如下：

Call <过程名> ([<实参表>)]

直接使用过程名，语法格式如下：

<过程名> [

**【说明】**

① 当用 Call 语句调用过程时，其过程名后必须加括号，若有参数，则参数必须放在括号之内。

② 若省略 Call 关键字，则过程名后不能加括号，若有参数，则参数直接跟在过程名之后，参数与过程名之间用空格隔开，参数与参数之间用逗号分隔。

例如，下面两个语句都能调用 shuru 子过程：

Call shuru(100)

shuru 100

其中的 100 是实际参数。实际参数可以是常量、变量或表达式，如：

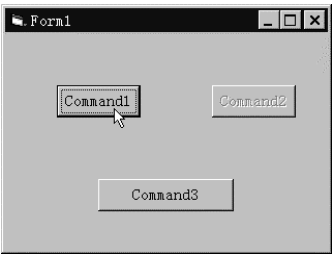
a=100

Call shuru(a)

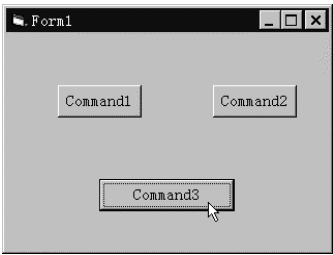
每次调用过程都会执行 Sub 和 End Sub 之间的<语句组>。子过程以 Sub 开始，以 End Sub 结束。当程序遇到 End Sub 时，退出过程，立即返回调用语句的后续语句。

### 8.2.3 子过程使用示例

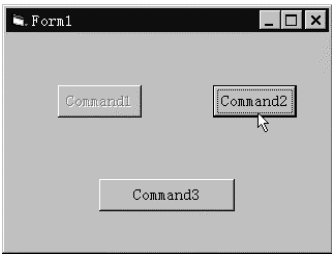
**【例 8-1】** 在一个窗口中包含 3 个命令按钮，当用户单击其中一个时，要求其他个别按钮不能使用，如图 8-5 所示。



(a) 单击 Command1 按钮后



(b) 单击 Command3 按钮后



(c) 单击 Command2 按钮后

图 8-5 3 个按钮

**【分析】** 本例可以分别建立 3 个按钮的 Click 事件过程，也可以建立一个通用子过程来处理 3 个命令按钮的 Click 事件。假设，单击 Command1 按钮使 Command2 按钮不可用，单击 Command2 按钮使 Command1 按钮不可用，单击 Command3 按钮使得 Command1 按钮和 Command2 按钮都可用。

图 8-6 说明了通用过程的使用过程，Click 事件中的代码调用按钮管理子过程，子过程运行自身的代码，然后将控制返回 Click 事件过程。

应用程序用户界面的建立与对象属性的设置，如图 8-5 所示。过程代码的编写步骤如下。

(1) 双击窗体的空白区，打开代码编辑窗口。执行“工具”→“添加过程”菜单命令，打开“添加过程”对话框。在“名称”文本框中输入过程名“Button\_Manage”，从“类型”组中选择“子程序”项，从“范围”组中选择“公有的”项，如图 8-7（a）所示。单击“确定”按钮后，在代码窗口中可以看到添加了一个子过程，如图 8-7（b）所示。

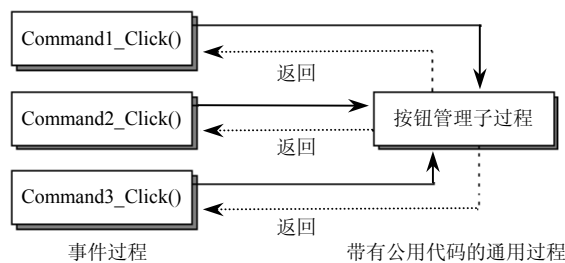


图8-6 通过程被事件过程调用的过程示意图

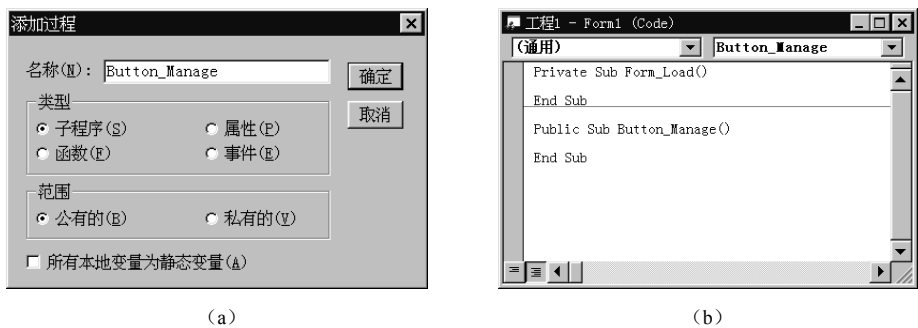


图 8-7 “添加过程”对话框与代码编辑窗口

(2) 通用过程代码如下：

```
Public Sub Button_Manage(Button As Object)
    Select Case Button
        Case Command1
            Command2.Enabled = False
        Case Command2
            Command1.Enabled = False
        Case Command3
            Command1.Enabled = True
            Command2.Enabled = True
    End Select
End Sub
```

代码编辑窗口如图 8-8(a) 所示。然后分别双击 3 个命令按钮，编写 3 个命令按钮的 Click 事件过程，如图 8-8（b）所示。

(3) 运行工程。单击标准工具栏中的“启动”按钮，运行工程，单击 Command1 按钮，窗体显示如图 8-5(a)所示。单击 Command3 按钮，窗体显示如图 8-5(b)所示。单击 Command2 按钮，窗体显示如图 8-5（c）所示。

【例 8-2】 分别计算阶乘  $5!$ ,  $6!$ ,  $8!$  及阶乘的和  $5! + 6! + 8!$ , 界面如图 8-9 所示。

【分析】要计算  $s = 5! + 6! + 8!$ , 先要分别计算出  $5!$ ,  $6!$  和  $8!$ 。由于 3 个求阶乘的运算过程完全相同, 因此可以用通用子过程来计算任意阶乘  $\text{tot}!$ 。每次调用子过程前给  $\text{tot}$  变量一个值, 在子过程中将所求结果放入  $\text{total}$  变量中, 返回主程序后,  $\text{tot}$  变量接收  $\text{total}$  的值。这样 3 次调用子程序便可求得变量  $s$ 。

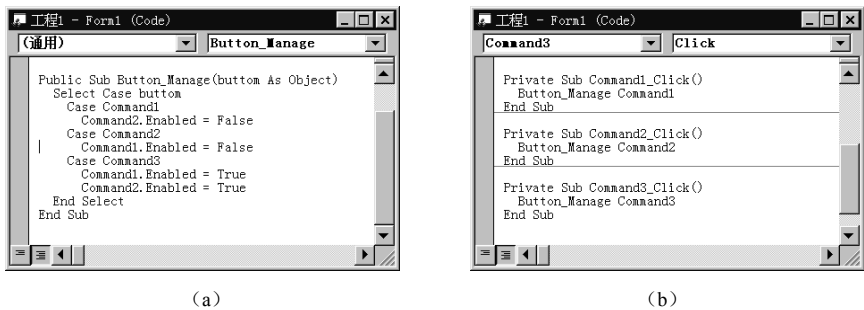


图 8-8 编写子过程代码与命令按钮的事件过程代码

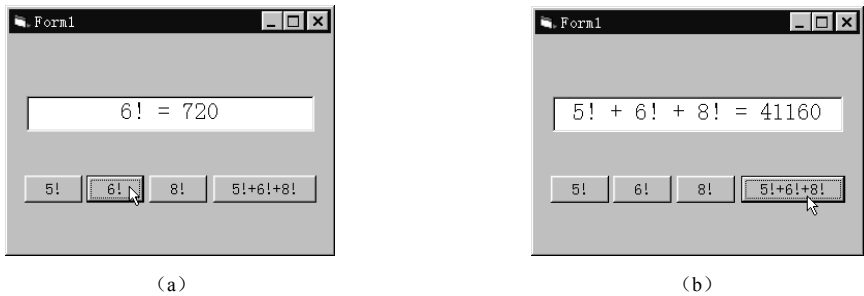


图 8-9 计算阶乘及阶乘的和

设计步骤如下。

应用程序用户界面的建立与对象属性的设置参见图 8-9。下面介绍过程代码的编写步骤。

(1) 双击窗体的空白区, 打开代码编辑窗口。执行“工具”→“添加过程”菜单命令, 打开“添加过程”对话框。在“名称”文本框中输入过程名“fact”, 从“类型”组中选择“子程序”项, 从“范围”组中选择“公有的”项, 如图 8-10 (a) 所示。单击“确定”按钮后, 在代码窗口中可以看到添加了一个子过程, 如图 8-10 (b) 所示。

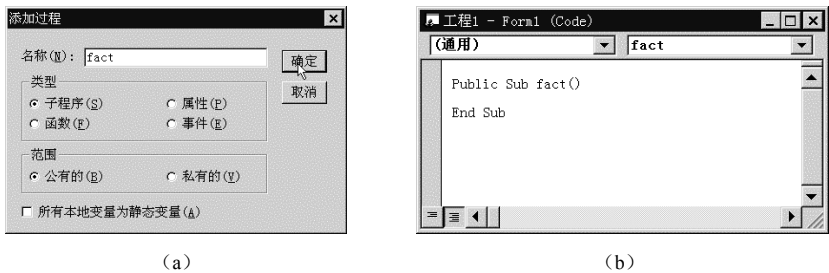


图 8-10 “添加过程”对话框与代码编辑窗口

(2) 编写通用过程代码。

在括号中添加形参表“ $m$  As Integer,  $\text{total}$  As Long”,  $\text{fact}$  通用子过程代码为:

```

Sub fact(m As Integer, total As Long)           ' 计算阶乘子过程
    Dim i As Integer
    total = 1
    For i = 1 To m
        total = total * i
    Next i
End Sub

```

(3) 编写事件过程来调用通用过程。

命令按钮组的 Click 事件代码为：

```

Private Sub Command1_Click(Index As Integer)
    Dim a As Integer, b As Integer, c As Integer, s As Long, tot As Long
    n = Index
    Select Case n
        Case 0
            a = 5
            Call fact(a, tot)
            Label1.Caption = a & "!=" & tot
        Case 1
            a = 6
            Call fact(a, tot)
            Label1.Caption = a & "!=" & tot
        Case 2
            a = 8
            Call fact(a, tot)
            Label1.Caption = a & "!=" & tot
        Case 3
            a = 5: b = 6: c = 8
            Call fact(a, tot)
            s = tot
            Call fact(b, tot)
            s = s + tot
            Call fact(c, tot)
            s = s + tot
            Label1.Caption = a & "!" + " & b & "!" + " & c & "!" = " & s
    End Select
End Sub

```

**【例 8-3】** 利用子过程编写计算圆面积的程序。

**【分析】** 利用子过程计算并输出圆面积，它有一个参数，即圆半径  $r$ 。在按钮的 Click 事件过程中，从文本框中接收半径值作为实参调用子过程。

设计步骤如下。

(1) 建立用户界面并设置对象属性，如图 8-11 所示。

(2) 编写事件代码。

在代码窗口中直接编写输入通用子过程代码：

```
Sub cir(r)
    Const pi = 3.14
    s = pi * r ^ 2
    MsgBox "圆面积为：" & s
```

End Sub

编写“计算圆面积”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()
    bj = Val(Text1.Text)
    Call cir(bj)
End Sub
```



图 8-11 计算圆面积

### 8.3 函数过程

VB 中包含了许多内部函数，如 Int、Sqr 等。用户在编写程序时，只需写出一个函数名并给定参数就能得出函数值。但是，如果用户在程序中需要多次用到某一公式或处理某一函数关系，而又没有现成的内部函数可用时，用户可以自己编写函数（Function）过程。函数过程与内部函数一样，可以在程序或函数嵌套中使用。

#### 8.3.1 定义函数过程

与子过程一样，函数过程也是一个独立的过程，可读取参数，执行一系列语句并改变其参数的值。与子过程不同的是，函数过程可返回给调用过程一个值。

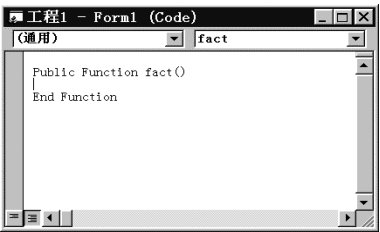
函数过程的定义与子过程相似，可以使用“添加过程”对话框，也可以在代码编辑窗口中直接输入过程代码。

##### 1. 使用“添加过程”对话框

例如，创建一个用于求某数阶乘的通用函数 fact，打开“添加过程”对话框，在“名称”文本框中输入过程名“fact”，在“类型”组中选择“函数”项，如图 8-12（a）所示。单击“确定”按钮，即可得到如图 8-12（b）所示的函数过程的框架。



(a)



(b)

图 8-12 定义函数过程

通常，由系统自动产生的函数过程框架还需要进行适当修改。由于函数过程有返回值，这个值就应该属于某种数据类型，因此，还需要在过程名后面加上对其返回值类型的定义和说明。另外，为了获得传递过来的参数，还需定义接收参数的变量，等等。

2. 在代码编辑窗口中输入

定义函数过程的语法格式为：

```
[ Private | Public ][ Static ] Function  〈函数名〉 ([ 〈形参表〉 ] ) [ As  〈类型〉 ]  
    [ 〈语句组〉 ]  
    [ 〈函数名〉 = 〈表达式〉 ]  
    [ Exit Function ]  
    [ 〈语句组〉 ]  
    [ 〈函数名〉 = 〈表达式〉 ]  
  
End Function
```

【说明】

①〈表达式〉的值是函数返回的结果。在语法中通过赋值语句将值赋给〈函数名〉，该值就是函数过程返回的值。

如果在函数过程中省略“〈函数名〉 = 〈表达式〉”，则该过程返回一个默认值：数值函数过程返回 0，字符串函数过程返回空字符串。因此，为了能使一个函数过程完成所指定的操作，通常要在过程中为〈函数名〉赋值。

②〈语句组〉中可以用一个或多个 Exit Function 语句从函数过程中退出。

③ 其他部分的含义与子过程相同。

【例 8-4】 编写计算圆面积的函数过程。

代码如下：

```
Function cir(r As Single) As Single  
    Const pi = 3.14                ' 定义符号常量  
    cir = pi * r ^ 2                ' 为函数名 cir 赋值  
  
End Function
```

8.3.2 调用函数过程

1. 直接调用

函数过程的调用很简单，与使用 VB 内部函数一样，可以在表达式中直接写上它的名字。例如，假设已经编好计算圆面积的函数过程 cir()，调用方法可为：

```
MsgBox "圆面积为：" & cir(10)
```

2. 用 Call 语句调用

与调用子过程一样调用函数过程。利用下面的代码可以调用同一个函数过程：

```
Call cir (10)  
  
area 10
```

当用这种方法调用函数时，VB 放弃返回值。



### 3. 无参函数的调用

函数可以没有参数，在调用无参函数时不发生虚实结合。调用无参函数得到一个固定的值，如下述无参函数：

```
Function a
    a = "ABCD"
End Function
```

可用如下方法调用：

```
Print a
```

### 8.3.3 函数过程使用示例

【例 8-5】 利用函数过程求圆面积。  
界面及属性设置参见例 8-3。代码如下：

```
Private Sub Command1_Click()
    Dim bj As Single
    bj = Val(Text1.Text)
    MsgBox "圆面积为：" & cir(bj) ' 调用函数过程
End Sub
```

```
Function cir(r As Single) As Single
    Const pi = 3.14
    cir = pi * r ^ 2 ' 返回函数值
End Function
```

【例 8-6】 编写求两个数中较大数的函数过程，并利用该函数过程求 3 个数中的最大数。

设计步骤如下。

- （1）建立用户界面并设置对象属性，如图 8-13 所示。
- （2）编写代码。

编写函数过程，返回两个数中的较大数：

```
Private Function Max(x As Single, y As Single)
    If x > y Then
        Max = x
    Else
        Max = y
    End If
End Function
```

编写“求最大数”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()
    Dim a As Single, b As Single, c As Single
```

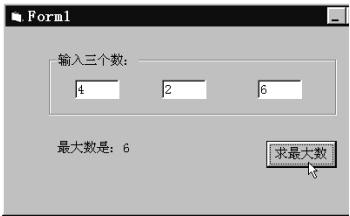


图 8-13 求 3 个数中的最大数

```

a = Val(Text1.Text)
b = Val(Text2.Text)
c = Val(Text3.Text)
Label1.Caption = "最大数是: " & Max(a, Max(b, c))      ' 调用函数过程

```

**End Sub**

【例 8-7】 编写判断某数是否能同时被 3, 5, 7 整除的函数过程，并输出 1~1000 之间所有能同时被 3, 5, 7 整除的数。

设计步骤如下。

(1) 建立用户界面并设置对象属性，参见图 8-14 (a)。

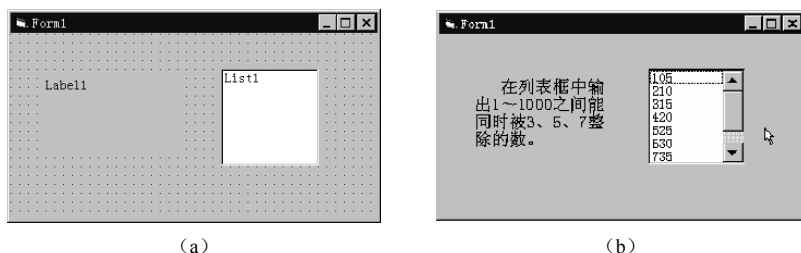


图 8-14 求能同时被 3, 5, 7 整除的数

(2) 编写代码。

编写函数过程代码：

```

Private Function D(x As Integer) As Boolean
    If (x Mod 3 = 0) And (x Mod 5 = 0) And (x Mod 7 = 0) Then
        D = True
    Else
        D = False
    End If

```

**End Function**

编写窗体 Form 的 Activate (控件激活) 事件代码：

```

Private Sub Form_Activate()
    Dim n As Integer
    For n = 1 To 1000                                ' 数据范围
        If D(n) Then                                  ' 调用函数过程进行判断
            List1.AddItem n                            ' 添加项目
        End If
    Next

```

**End Sub**

运行程序，结果如图 8-14 (b) 所示。

【例 8-8】 求 3~10 的阶乘之和，界面如图 8-15 所示。

【分析】 利用例 8-2 中求阶乘的函数过程 fact。主程序通过调用该函数依次求得 3!, 4!, 5!, ..., 10! 的值，然后把这些值相加。

应用程序用户界面的建立与对象属性的设置，参见图 8-15。  
函数过程的代码参见例 8-2，下面给出“计算”命令按钮的 Click 事件代码：

```
Private Sub Command1_Click()  
    Dim sum As Long, i As Integer  
    For i = 3 To 10  
        sum = sum + fact(i)  
    Next i  
    Label2.Caption = sum  
End Sub
```

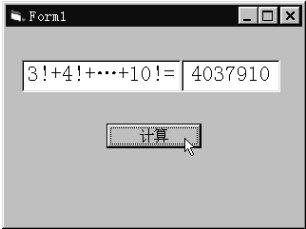


图 8-15 求 3~10 的阶乘之和

### 8.3.4 查看过程

#### 1. 查看当前模块中的过程

若要查看现有的通用过程或函数过程，可在代码编辑器窗口的对象列表框中选择“通用”项，然后在过程列表框中选择过程名。若要查看事件过程，可在代码编辑器窗口的对象列表框中选择适当的对象，然后在过程列表框中选择事件。



图 8-16 对象浏览器

#### 2. 查看其他模块中的过程

执行“视图”→“对象浏览器”菜单命令，打开窗口如图 8-16 所示。在工程/库列表框中选择工程，在类/模块列表框中选择模块，并在成员列表框中选择过程，单击“查看定义”按钮。

## 8.4 参数传递

调用过程的目的，就是在一定的条件下完成某一工作或计算某一函数值。外界需要把条件告诉过程，反过来，过程也需要把某些结果报告给外界，这就是过程与外界的数据传递。  
过程与外界的数据传递方式有以下两种：

- 通过非局部变量
- 通过参数

在过程体中使用非局部变量（如全程变量），就是直接处理外界的量。由于这种量在过程内、外都能用，因此数据传递不成问题。本节主要讨论参数的传递问题。

### 8.4.1 形式参数与实际参数

形式参数是在子过程和函数过程的定义中出现的变量名。实际参数则是在调用子过程和函数过程时，传送给子过程和函数过程的常数、变量、表达式或数组。  
在 VB 中，通常把形式参数叫做“形参”，把实际参数叫做“实参”。

1. 形参表

形参表中的各个变量之间用逗号分隔，表中的变量可以是：

- 后面跟有左、右圆括号的数组名
- 除定长字符串之外的合法变量名

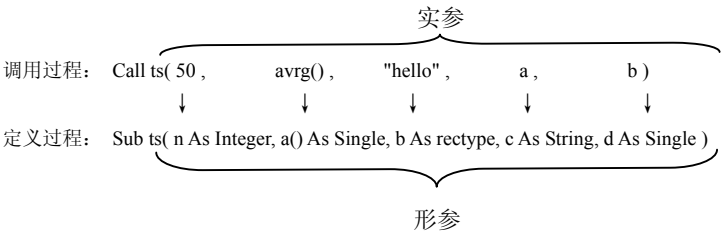
在形参表中只能用 `a As String` 之类的变长字符串作为形参，不能用 `a As String*8` 之类的定长字符串作为形参，但定长字符串可以作为实参传递给过程。

2. 实参表

实参表中的各项用逗号隔开，实参可以是：常量、表达式、合法的变量名或后面跟有左、右圆括号的数组名。

3. 形参与实参的对应关系

形参与实参的对应关系为：



【说明】

① 在定义过程时，形参为实参保留位置。在调用过程时，实参被插入形参中对应的各变量处，第一个形参接收第一个实参的值，第二个形参接收第二个实参的值……

② 实参表和形参表中对应的变量名不必相同，但是变量的个数必须相等，且各实参的类型必须与相应的形参相符。

8.4.2 按值传递与按地址传递

在调用过程时，一般调用过程与定义过程之间有数据传递，即将调用过程的实参传递给定义过程，完成实参与形参的结合，然后执行调用过程。

在 VB 中，实参与形参的结合有两种方法，即“传址”和“传值”。

传递参数的方式有两种：如果调用语句中的实际参数是常量或表达式，或者定义过程时选用 `ByVal` 关键字，就可以按值传递；如果调用语句中的实际参数为变量，或者定义过程时选用 `ByRef` 关键字，就可以按地址传递。

1. 传址

传址就是让过程根据变量的内存地址去访问实际变量的内容，即形参与实参使用相同的内存地址单元，这样通过子过程就可以改变变量本身的值。系统默认按地址传递参数。

在传址调用时，实参必须是变量，常量或表达式无法传址。

【例 8-9】 传址调用示例。

现在有下列的通用过程：

```
Sub try(x As Integer, y As Integer)
    x = x + 2           ' 在子程序中改变变量的值
    y = y + 3           ' 在子程序中改变变量的值
    Print "x="; x, "y="; y ' 在子程序中输出变量的值
End Sub
```

窗体 Form 的 Click 事件代码如下：

```
Private Sub Form_Click()
    Dim a As Integer, b As Integer
    a = 5                 ' 在主程序中变量的原值
    b = 6                 ' 在主程序中变量的原值
    try a, b              ' 传址调用
    Print "a="; a, "b="; b ' 在主程序中输出变量的值
End Sub
```

在 Click 事件过程中，通过 try a, b 语句调用过程 try，实参 a 和 b 的值分别为 5 和 6，传送给 try 后进行计算，在通用过程中输出的 x 和 y 分别为 7 和 9，而在事件过程中输出的 a 和 b 同样为 7 和 9。因此，运行上述程序后，输出结果如图 8-17 所示。

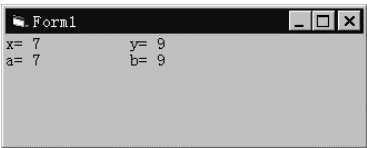


图 8-17 传址调用示例

2. 传值

传值就是通过值传送实参，即传送实参的值，而不是传送它的地址。在这种情况下，系统把需要传送的变量复制到一个临时单元中，然后把该临时单元的地址传送给被调用的通用过程。由于通用过程没有访问变量（实参）的原始地址，因而不会改变原来变量的值，所有的变化都是在变量的副本上进行的。

当要求变量按值传送时，可以用下面的方法。

- 把变量变成一个表达式。把变量转换成表达式的最简单的方法就是把它放在括号内，例如，把变量用括号括起来，把它变为一个表达式，如“(a)”。
- 定义过程时用 ByVal 关键字指出参数是按值传递的，例如：

```
Sub PostAc( ByVal x As Integer )
    x = x + 2
End Sub
```

这里的形参 x 前有关键字 ByVal，调用时以传值方式传送实参。在传值方式下，VB 为形参分配内存空间，并将相应的实参值复制给各形参。

【例 8-10】 传值调用示例。将例 8-9 改用传值方式编写通用过程，则运行结果将不同。改为传值方式的通用过程如下：

```
Sub try(ByVal x As Integer, ByVal y As Integer)
    x = x + 2           ' 在子程序中改变变量的值
    y = y + 3           ' 在子程序中改变变量的值
```

```
Print "x="; x, "y="; y
```

' 在子程序中输出变量的值

**End Sub**

窗体 Form 的 Click 事件代码与例 8-9 相同:

**Private Sub Form\_Click()**

```
Dim a As Integer, b As Integer
```

```
a = 5
```

' 在主程序中变量的原值

```
b = 6
```

' 在主程序中变量的原值

```
try a, b
```

' 传值调用

```
Print "a="; a, "b="; b
```

' 在主程序中输出变量的值

**End Sub**

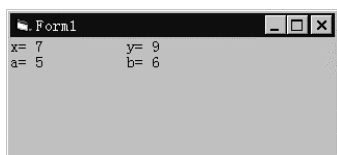


图 8-18 传值调用示例

运行程序后, 输出结果如图 8-18 所示。

过程中的变量名即使与主程序中使用的变量名相同, 在内存中也占用不同的内存单元地址。在执行过程时, 即使其变量内容发生变化, 主程序中的变量内容也不会随之改变。

用传值调用这种传递参数的方法只能传递计算值, 如数值、字符串。

采用值参数 (传值) 只能从外界向过程 (函数) 传入信息, 但不能传出; 而采用变量参数 (传址) 则既能传入, 又能传出。正是由于不能传出, 过程结束后, 值参数的值就不会影响外界的任何量, 因而在一定意义上说, 值参数比较安全。

变量参数和值参数各有特点, 采用哪一种更合适, 则需要视情况而定。一般来说, 需要传出参数值时应该用变量参数, 否则采用值参数较好。

值参数与变量参数的一种重要区别是, 值参数对应的实参是表达式, 而变量参数对应的只能是变量。

## 8.4.3 使用参数

### 1. 使用可选的参数

在过程的形参表中加上 **Optional** 关键字, 就可以指定过程的形式参数为可选的。如果指定了可选参数, 则参数表中此参数后面的其他参数也必须是可选的, 并且每个参数都要用 **Optional** 关键字来声明。

**【例 8-11】** 两个命令按钮的事件代码调用同一个过程, 一个命令按钮传递两个参数, 而另一个命令按钮传递一个参数, 如图 8-19 所示。

子过程 **ListText** 将传递过来的参数值添加到列表框中, 其中第二个参数使用 **Optional** 关键字来声明:

**Private Sub ListText(x As String, Optional y As String)**

```
If IsMissing(y) Then
```

```
temp = Format(x, "@@@@@@@@@@")
```

```
Else
```

```
temp = Format(x, "@@@@@@@@@@") & Format(y, "@@@@@@@@@@")
```



图 8-19 使用可选参数

```
End If
List1.AddItem temp
```

**End Sub**

传递两个参数的命令按钮 Command1 的 Click 事件代码:

```
Private Sub Command1_Click()
    Dim a As String, b As String
    a = InputBox("")
    b = InputBox("")
    Call ListText(a, b)

```

**End Sub**

传递一个参数的命令按钮 Command2 的 Click 事件代码:

```
Private Sub Command2_Click()
    Dim a As String
    a = InputBox("")
    Call ListText(a)

```

**End Sub**

【说明】在未提供某个可选参数时，实际上将该参数作为具有 Empty 值的变量来赋值。上例说明了如何用 IsMissing 函数测试丢失的可选参数。

2. 提供可选参数的默认值

可以给可选参数指定默认值。

【例 8-12】 在上例中，可以为未传递的参数指定一个默认值，如图 8-20 所示。只需修改子过程 ListText 的代码:

```
Private Sub ListText(x As String, Optional y As String = "*****")
    temp = Format(x, "@@@@@@@@@@@@@@") & Format(y, "@@@@@@@@@@@@@@")
    List1.AddItem temp

```

**End Sub**

3. 使用不定数量的参数

一般来说，过程调用中的参数个数应等于过程说明中的参数个数。如果使用 ParamArray 关键字，则过程可以接收任意个数的参数。

【例 8-13】 改写例 8-11 中的子过程 ListText，使之可以接收任意多个参数，如图 8-21 所示。



图 8-20 可选参数的默认值



图 8-21 接收任意多个参数

子过程 ListText 的代码改为:

```
Private Sub ListText(ParamArray strx())  
    For Each x In strx  
        temp = temp & Format(x, "@@@@@@")  
    Next  
    List1.AddItem temp  
End Sub
```

传递三个参数的命令按钮 Command1 的 Click 事件代码:

```
Private Sub Command1_Click()  
    Dim a As String, b As String, c As String  
    a = Left(InputBox(""), 6)  
    b = Left(InputBox(""), 6)  
    c = Left(InputBox(""), 6)  
    Call ListText(a, b, c)  
End Sub
```

传递两个参数的命令按钮 Command2 的 Click 事件代码:

```
Private Sub Command2_Click()  
    Dim a As String, b As String  
    a = Left(InputBox(""), 6)  
    b = Left(InputBox(""), 6)  
    Call ListText(a, b)  
End Sub
```

#### 8.4.4 传递数组

在使用通用过程和函数过程时, 可以将数组或数组元素作为参数进行传递。传递整个数组时, 在实参与其所对应的形参都必须写上所要传递的数组的名称和一对圆括号, 如 arr()。在子程序中不可再用 Dim 语句来定义所要传递的数组。

如果要传递数组中的某一元素, 则在 Call 语句中只需直接写上该数组元素, 例如:

```
Call test(5,x(3))
```

【例 8-14】 在例 7-7 中编写排序子过程, 将存有随机数的数组作为参数传递到子过程中。

窗体界面与例 7-7 相同, 增加一个子过程 pai\_xu(), 并且改写“排序”按钮的 Click 事件代码, 其余过程代码不变。

子过程 pai\_xu() 的代码如下:

```
Sub pai_xu(p() As Integer) ' 如果形参定义了类型, 实参也必须定义相同的类型  
    For i = 1 To 9  
        For j = i + 1 To 10  
            If p(i) > p(j) Then  
                t = p(i): p(i) = p(j): p(j) = t
```



```

End If
Next
Next
End Sub
“排序”按钮 Command2 的 Click 事件代码如下：
Private Sub Command2_Click()
    p = ""
    Call pai_xu(a())
    For i = 1 To 10
        p = p & Str(a(i)) & ", "
    Next
    Label2.Caption = LTrim(Left(p, Len(p) - 1))
End Sub

```

在模块的通用段定义了与形式参数相同数据类型的数组：

```
Dim a(1 To 10) As Integer
```

### 【说明】

- ① 程序的运行结果与例 7-7 是相同的。
- ② 数组传递是“地址传递”，实际传递的是数组首元素的地址，因此对子过程的调用改变了数组中的原有数据。

**【例 8-15】** 随机产生 3 组整数（两位数），每组 10 个数，并求出每组的最大数，界面如图 8-22 所示。

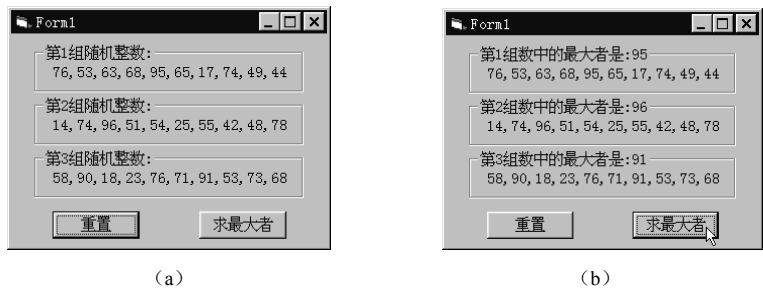


图 8-22 分别求 3 组随机整数中的最大数

**【分析】** 由于要多次产生随机整数数组并求出其最大数，因此需要编写产生随机整数数组的函数过程及求数组中最大整数的函数过程，在主程序中调用并向函数过程传递数组。设计步骤如下。

(1) 设计窗体界面并设置对象属性。新建一个工程，进入窗体设计器。首先在窗体中增加两个命令按钮 Command1, Command2 和一个框架 Frame1。选中 Frame1，在其中增加一个标签 Label1。然后将框架和其中的标签复制为控件数组：Frame1(0)~Frame1(2)和 Label1(0)~Label1(2)。

各控件属性的设置如图 8-22 所示。

(2) 编写程序代码。

考虑到要在不同的过程中使用数组，所以首先在模块的通用段声明数组：

```
Dim a(1 To 10) As Integer, b(1 To 10) As Integer, c(1 To 10) As Integer
```

产生随机数组并将 10 个数连成一个字符串返回的函数过程代码如下：

```
Function sui_ji(p() As Integer) As String
```

```
temp = ""
```

```
Randomize
```

```
For i = 1 To 10
```

```
Do
```

```
    x = Int(Rnd * 90) + 10
```

```
    yes = 0
```

```
    For j = 1 To i - 1
```

```
        If x = p(j) Then yes = 1: Exit For
```

```
    Next
```

```
    Loop While yes = 1
```

```
    p(i) = x
```

```
    temp = temp & LTrim(Str(p(i))) & ","
```

```
Next
```

```
sui_ji = temp
```

```
End Function
```

找出数组 10 个数中最大数并返回的函数过程代码如下：

```
Function da(p() As Integer) As Integer
```

```
Max = p(1)
```

```
For i = 2 To 10
```

```
    If Max < p(i) Then Max = p(i)
```

```
Next
```

```
da = Max
```

```
End Function
```

随机整数的生成由窗体的 Load 事件代码完成：

```
Private Sub Form_Load()
```

```
Dim p As String
```

```
p = sui_ji(a())
```

```
Frame1(0).Caption = "第 1 组随机整数:"
```

```
Label1(0).Caption = LTrim(Left(p, Len(p) - 1))
```

```
p = sui_ji(b())
```

```
Frame1(1).Caption = "第 2 组随机整数:"
```

```
Label1(1).Caption = LTrim(Left(p, Len(p) - 1))
```

```
p = sui_ji(c())
```

```
Frame1(2).Caption = "第 3 组随机整数:"
```

```
Label1(2).Caption = LTrim(Left(p, Len(p) - 1))
```

```
End Sub
```

两个命令按钮的 Click 事件代码如下：

```
Private Sub Command1_Click()
```

```
Form_Load
```

```
End Sub
```

```
Private Sub Command2_Click()
```

```
Dim p As String
```

```
p = "第 1 组数中的最大者是:"
```

```
Frame1(0).Caption = p & da(a())
```

```
p = "第 2 组数中的最大者是:"
```

```
Frame1(1).Caption = p & da(b())
```

```
p = "第 3 组数中的最大者是:"
```

```
Frame1(2).Caption = p & da(c())
```

```
End Sub
```

## 8.5 过程的嵌套与递归调用

在一个过程（子过程或函数过程）中调用另外一个过程，称为过程的嵌套调用；而过程直接或间接地调用其自身，则称为过程的递归调用。

### 8.5.1 过程的嵌套调用

VB 的过程定义都是互相平行和孤立的，也就是说，在定义过程时，一个过程内不能包含另一个过程。VB 虽然不能嵌套定义过程，但可以嵌套调用过程，也就是说，主程序可以调用子过程，在子过程中还可以调用另外的子过程，这种程序结构称为过程的嵌套。如图 8-23 所示。

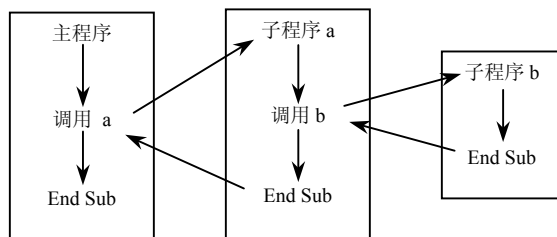


图 8-23 过程的嵌套

从图 8-23 可以看出，主程序或子过程遇到调用子过程语句就转去执行子过程，而本程序的余下部分要等从子过程返回后才得以继续执行。

**【例 8-16】** 输入参数  $n, m$ ，求组合数  $C_n^m = \frac{n!}{m!(n-m)!}$  的值。

**【分析】** 求组合数用函数过程 Comb 来实现，求阶乘  $n!$  则由另一个函数过程 fact 来实

现。在执行 Comb 函数的过程中要多次调用 fact 函数，即嵌套调用过程。  
设置步骤如下。

- (1) 建立用户界面并设置对象属性，如图 8-24 (a) 所示。
- (2) 编写代码。

求阶乘的函数过程 fact 的代码如下：

```
Private Function fact(x)
    p = 1
    For i = 1 To x
        p = p * i
    Next i
    fact = p ' 返回函数值
End Function
```

求组合数的函数过程 Comb 的代码如下：

```
Private Function comb(n, m)
    comb = fact(n) / (fact(m) * fact(n - m)) ' 计算并返回函数值
End Function
```

“计算组合数” 命令按钮 Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()
    m = Val(Text1.Text)
    n = Val(Text2.Text)
    If m > n Then
        MsgBox "输入数据不正确！ ", 0, "请检查！ " ' 数据检验
        Exit Sub ' 退出本过程
    End If
    Label2.Caption = "组合数是： " & comb(n, m)
End Sub
```

运行程序，在文本框中分别输入参数 n 和 m，输出组合数结果，如图 8-24 (a) 所示。  
如果 n>m，则弹出消息框，提示输入数据不正确，如图 8-24 (b) 所示。

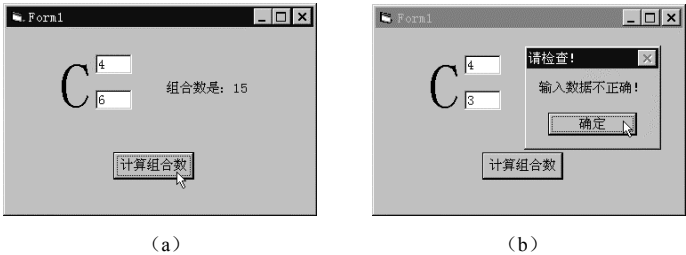


图 8-24 求组合数

8.5.2 过程的递归调用

递归算法是指一个过程直接或间接调用自己本身，即自己调用自己。递归在算法描述中

有着不可替代的作用。很多看似十分复杂的问题，使用递归算法来描述就显得非常简捷与清晰了。

递归调用在处理阶乘运算、级数运算、幂指数运算等方面特别有效。  
在递归调用中，一个过程执行的某一步要用到它自身的上一步（或上几步）的结果。

【例 8-17】 利用递归调用计算  $n!$ ，界面如图 8-25 所示。

【分析】自然数  $n$  的阶乘可以递归定义为：

$$n! = \begin{cases} 1 & n = 0 \\ n \times (n-1)! & n > 0 \end{cases}$$

- (1) 窗体的设计及对象属性的设置，参见图 8-25。
- (2) 编写代码。下面给出函数过程与事件过程的代码。

求阶乘的递归函数过程 fact 的代码如下：

```
Private Function fact(n) As Double
    If n > 0 Then
        fact = n * fact(n - 1)
    Else
        fact = 1
    End If
End Function
```

文本框的 KeyPress 事件代码：

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
    Dim n As Integer, m As Double
    If KeyAscii = 13 Then
        n = Val(Text1.Text0)
        If n < 0 Or n > 20 Then MsgBox ("非法数据！"): Exit Sub
        m = fact(n)
        Text2.Text = Format(m, "!@@@@@@@@@@@@")
        Text1.SetFocus
    End If
End Sub
```

【说明】当  $n > 0$  时，在 fact 过程中调用 fact 过程，参数为  $n - 1$ ，这种操作一直持续到  $n = 1$  为止。

例如，当  $n = 5$  时，求 fact(5) 的值变为求  $5 \times \text{fact}(4)$ ；求 fact(4) 的值又变为求  $4 \times \text{fact}(3)$ ，……，当  $n = 0$  时，fact 的值为 1，递归结束，其结果为  $5 \times 4 \times 3 \times 2 \times 1$ 。如果把第一次调用过程 fact 叫做 0 级调用，以后每调用一次过程，递归级别增加 1，过程参数  $n$  减 1，则递归调用的过程如下：

递归级别	执行操作
0	fact(5)
1	fact(4)
2	fact(3)
3	fact(2)

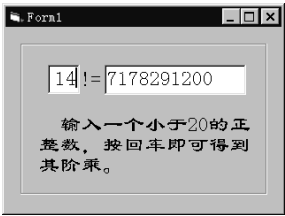


图 8-25 求阶乘

```

4                                fact(1)
4                                返回 1 fact(1)
3                                返回 2 fact(2)
2                                返回 6 fact(3)
1                                返回 24 fact(4)
0                                返回 120 fact(5)

```

【例 8-18】 利用递归过程编写程序打印 Fibonacci 数列。Fibonacci 数列为：

1 1 2 3 5 8 13 21 34 55 ...

【分析】形成此数列的规律为，头两个数为 1，从第 3 个数开始其值是它前面的两个数之和，即

$$\text{fibo} = \begin{cases} 1 & n = 1 \\ 1 & n = 2 \\ \text{fibo}(n-1) + \text{fibo}(n-2) & n > 2 \end{cases}$$

设计步骤如下。

(1) 建立用户界面并设置对象属性，如图 8-26 (a) 所示。

(2) 编写代码。

编写窗体 Form 的 Load 事件代码：

```

Private Sub Form_Load()
    n = InputBox("您需要输出的个数: ", "斐波那契数列")
    For x = 1 To n
        List1.AddItem fibo(x)           ' 在列表框中添加项目
    Next x
End Sub

```

编写函数过程：

```

Private Function fibo(n)
    If n = 1 Or n = 2 Then
        fibo = 1
    Else
        fibo = fibo(n - 1) + fibo(n - 2)   ' 递归调用
    End If
End Function

```

运行程序，结果如图 8-26 (b) 所示。

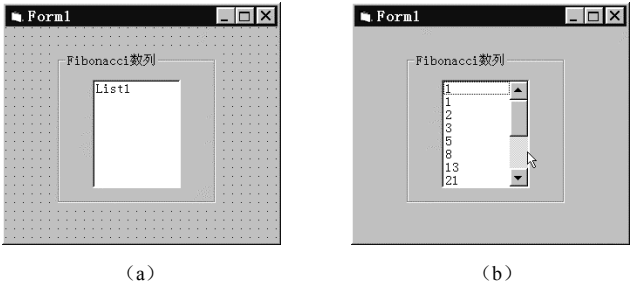


图 8-26 用户界面和运行结果

## 习题 8

### 一、选择题

8.1 在 VB 工程中, 可以作为“启动对象”的程序是 ( )。

- A) 任何窗体或标准模块                      B) 任何窗体或过程  
C) Sub Main 过程或其他任何模块              D) Sub Main 过程或任何窗体

8.2 以下关于函数过程的叙述中, 正确的是 ( )。

- A) 函数过程形参的类型与函数返回值的类型没有关系  
B) 在函数过程中, 过程的返回值可以有多个  
C) 当数组作为函数过程的参数时, 既能以传值方式传递, 也能以传址方式传递  
D) 如果不指明函数过程参数的类型, 则该参数没有数据类型

8.3 下列程序的执行结果为 ( )。

```
Private Sub Command1_Click()
```

```
    Dim x As Integer, y As Integer
```

```
    x=12 : y=20
```

```
    Call Value(x, y)
```

```
    Print x ; y
```

```
End Sub
```

```
Private Sub Value(ByVal m As Integer, ByVal n As Integer)
```

```
    m=m*2 : n=n-5
```

```
    Print m ; n
```

```
End Sub
```

- A) 20    12                      B) 12    20                      C) 24    15                      D) 24    12  
     20    15                      12    25                      12    20                      12    15

8.4 在窗体上画一个名称为 Text1 的文本框, 一个名称为 Command1 的命令按钮, 然后编写如下事件过程和通用过程:

```
Private Sub Command1_Click()
```

```
    n=Val(Text1.Text)
```

```
    If n\2=n/2 Then
```

```
        f=f1(n)
```

```
    Else
```

```
        f=f2(n)
```

```
    End If
```

```
    Print f;n
```

```
End Sub
```

```
Public Function f1(ByRef x)
```

```
    x=x*x
```

```
    f1=x+x
```

```
End Function
```

**Public Function f2(ByVal x)**

    x=x\*x

    f1=x+x+x

**End Function**

程序运行后，在文本框中输入 6，然后单击命令按钮，窗体上显示的是（ ）。

- A) 72    36                      B) 108    36                      C) 72    6                      D) 108    6

8.5 在窗体上画一个名称为 Command1 的命令按钮，然后编写如下通用过程和命令按钮的事件过程：

**Private Function f(m As Integer)**

    If m Mod 2=0 Then

        f=m

    Else

        f=1

    End If

**End Function**

**Private Sub Command1\_Click()**

    Dim i As Integer

    s=0

    For i=1 TO 5

        s=s+f(i)

    Next

    Print s

**End Sub**

程序运行后，单击命令按钮，在窗体上显示的是（ ）。

- A) 11                      B) 10                      C) 9                      D) 8

8.6 下面程序段，运行后的结果是（ ）。

**Private Sub Command1\_Click()**

    Dim b%(1 To 4),i%,t#

    For i=1 To 4

        b(i)=i

    Next i

    t=Tof(b())

    Print "t=" ; t,

**End Sub**

**Function Tof(a()As Integer)**

    Dim t#, i%

    t=1

    For i=2 To UBound(a)

        t=t\*a(i)



Next i

Tof=t

**End Function**

A) t=18

B) t=24

C) t=30

D) t=32

8.7 单击按钮时，以下程序运行后的输出结果是（ ）。

**Private Sub Command1\_Click()**

Dim x As Integer,y As Integer,z As Integer

x=1:y=2:z=3

Call God(x,x,z)

Print x;x;z

Call God(x,y,y)

Print x;y;y

**End Sub**

**Private Sub God(x As Integer,y As Integer,z As Integer)**

x=3\*z+1

y=2\*z

z=x+y

**End Sub**

A) 6 6 12

B) 8 5 10

C) 9 6 12

D) 8 10 10

7 11 11

5 11 11

9 10 15

5 9 10

## 二、填空题

8.8 以下是一个计算矩形面积的程序，调用过程计算矩形面积。请将程序补充完整。

**Sub RecArea(L, W)**

Dim S As Double

S=L \* W

MsgBox "Total Area is" & Str(S)

**End Sub**

**Private Sub Command1\_Click()**

Dim M,N

M=InputBox("What is the L? ")

M=Val(M)

\_\_\_\_\_

N=Val(N)

\_\_\_\_\_

**End Sub**

8.9 在窗体上画一个名称为 Command1 的命令按钮，然后编写如下程序：

Option Base 1

**Private Sub Command1\_Click()**

```

    Dim a(10)As Integer
    For i=1 To 10
        a(i)=i
    Next
    Call swap ____
    For i=1 To 10
        print a(i);
    Next
End Sub
Sub swap(b()As Integer)
    n= ____
    For i=1 To n/2
        t=b(i)
        b(i)=b(n)
        b(n)=t
        ____
    Next
End Sub

```

上述程序的功能是，通过调用过程 `swap`，调换数组中数值的存放位置，即 `a(1)`与 `a(10)` 的值互换，`a(2)`与 `a(9)`的值互换，……，`a(5)`与 `a(6)`的值互换。请填补全程序。

8.10 设有以下函数过程：

```

Function fun(m As Integer)As Integer
    Dim k As Integer , sum As Integer
    sum=0
    For k=m To 1 Step-2
        sum=sum+k
    Next k
    fun=sum
End Function

```

若在程序中用语句 `s=fun(10)`调用此函数，则 `s` 的值为\_\_\_\_\_。

### 三、编程题

8.11 编写判断奇偶数的函数过程。输入一个整数，判断其奇偶性。

8.12 编写随机整数函数过程，产生 30 个 1~100 之间的随机数。

8.13 编写程序，实现英语单词或短语的加密/解密操作。加密/解密的基本原则是：把英语单词或短语中每个字符的 ASCII 码加上 2，使其变为另外一个字符。例如：把“ABCDE”中每个字符的 ASCII 码加 2，变为“CEDFG”，从而对原来的单词或短语“加密”。

8.14 利用子过程或函数过程，求 1~6 的阶乘之和。

8.15 编写判断素数的子过程或函数过程，验证哥德巴赫猜想：一个不小于 6 的偶数可

以表示为两个素数之和。例如： $6 = 3 + 3, 8 = 3 + 5, 10 = 3 + 7, \dots$ 。

8.16 编写判断某数是否能同时被 17 与 37 整除的函数过程，并输出 1000~2000 之间所有能同时被 17 与 37 整除的数。

8.17 使用 Timer 函数设计用来暂停指定时间（秒）的子过程。

8.18 编写计算阶乘的函数过程，利用  $e^x$  的近似公式计算  $e$  值（直到最后一项小于  $10^{-6}$  为止）。

$$e^x \approx 1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots + \frac{x^n}{n!}$$

8.19 编写函数过程返回指定字符、长度的字符串，实现在窗体上输出如图 8-27 所示的图形。

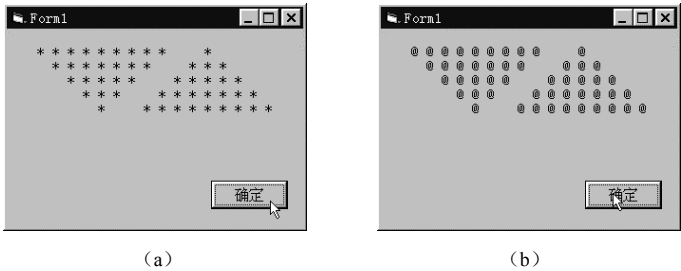


图 8-27 输出图形

8.20 有 5 个人坐在一起。问第 5 个人的岁数，他说比第 4 个人大 2 岁。问第 4 个人，他说比第 3 个人大 2 岁。问第 3 个人，说比第 2 个人大 2 岁。问第 2 个人，说比第 1 个人大 2 岁。最后问第 1 个人，他说是 10 岁。请问第 5 个人有多大岁数。

8.21 图 8-28 中各图分别由若干个大小不等、形状相同的三角形构成。形成该图的方法是从一个大的等边三角形开始，将其三条边的中点进行连线，分成相同的 4 个三角形，对除中间外的 3 个三角形再重复上述过程，直到到达满足给定条件的底层为止。

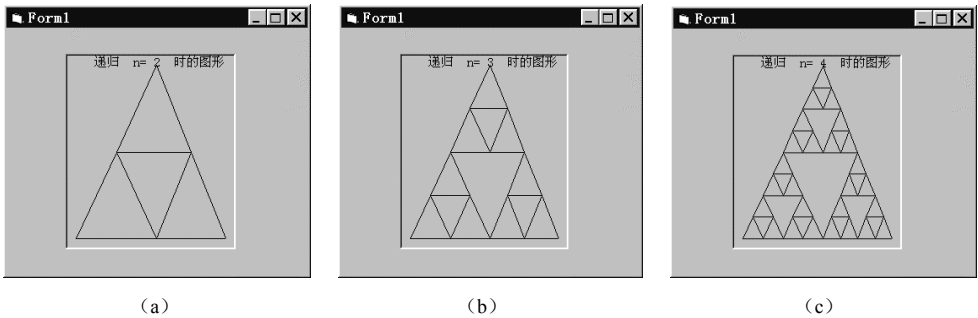


图 8-28 递归调用示例

# 第 9 章 变量与过程的作用域

在 VB 中，应用程序是由若干个过程组成的，这些过程一般保存在窗体文件（.frm）或标准模块文件（.bas）中。变量在过程中是必不可少的。根据变量或过程所处的不同位置，其可被访问的范围是不相同的。变量与过程可被访问的范围称为变量与过程的作用域。

## 9.1 代码模块的概念

对于一个大型的程序，在建立应用程序时，通常应先设计代码的组成结构。在一般情况下，VB 将代码存储在 3 种不同的模块中（窗体模块、标准模块和类模块），如图 9-1 所示。

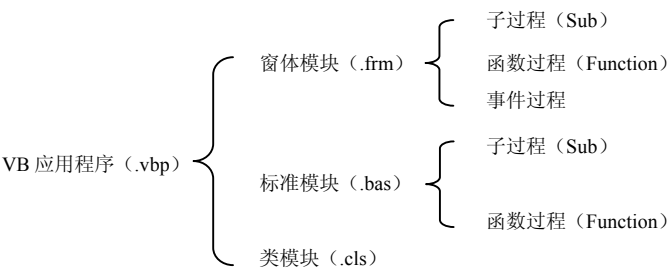


图 9-1 VB 应用程序的模块层次结构

在窗体模块、标准模块和类模块这 3 种模块中，都可以包含声明和过程。它们形成了工程的模块层次结构，这样可以较好地组织工程，也便于代码的维护。例如，某工程的模块结构如图 9-2 所示。

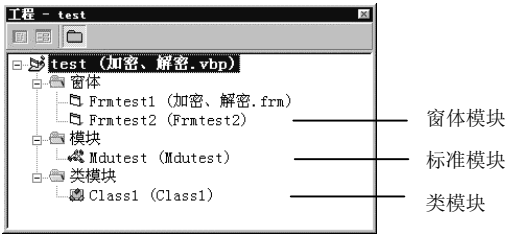


图 9-2 工程中的模块结构

### 1. 窗体模块

VB 中，每个窗体对应一个窗体模块，窗体模块包括窗体及其控件的属性设置、窗体变量的说明、事件过程、窗体内的通用过程、外部过程的窗体级声明。

窗体模块保存在扩展名为 .frm 的文件中。默认时，应用程序中只有一个窗体，因此只有一个以 .frm 为扩展名的窗体模块文件。如果应用程序有多个窗体，就会有多个以 .frm 为扩展名的窗体模块文件。

添加新窗体的步骤如下。

(1) 执行“工程”→“添加窗体”菜单命令，打开“添加窗体”对话框，如图 9-3 所示。

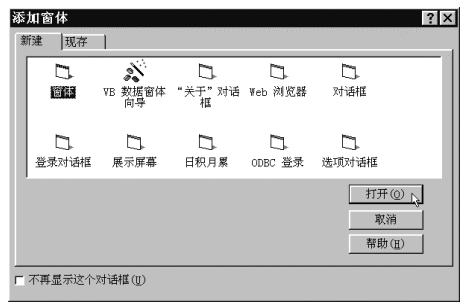


图 9-3 “添加窗体”对话框

(2) 在“添加窗体”对话框中选择“新建”选项卡，选中需要添加的窗体类型，单击“打开”按钮，新建的窗体将出现在工程窗口中。

注意：虽然 ActiveX 文档和 ActiveX 控件等是具有不同扩展名的新模块类型，但从编程角度来讲，这些模块仍可视作窗体模块。

2. 标准模块

标准模块可以包含公有或模块级的变量、常数、类型，外部过程和全局过程的全局声明或模块级声明。默认时，标准模块中的代码是公有的，任何窗体或模块中的事件过程或通用过程都可以调用它。

标准模块保存在扩展名为.bas 的文件中，默认时应用程序中不包含标准模块。

在工程中添加标准模块的步骤如下。

(1) 执行“工程”→“添加模块”菜单命令，打开“添加模块”对话框，选择“新建”选项卡，如图 9-4 (a) 所示。

(2) 双击“模块”图标，打开新建标准模块窗口，如图 9-4 (b) 所示。

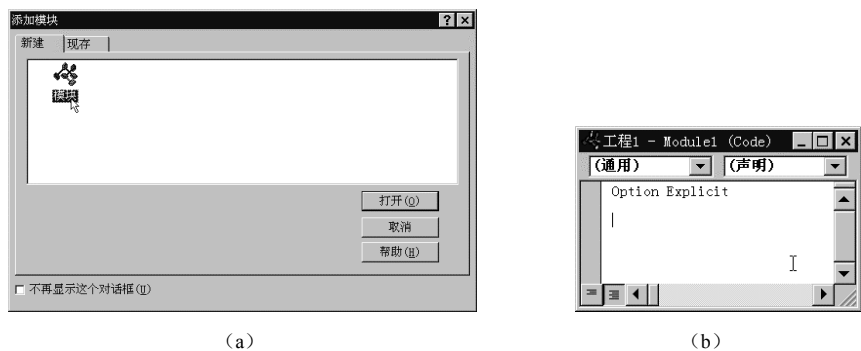


图 9-4 添加标准模块

(3) 在属性窗口中修改该模块的名称属性，给模块命名，然后就可在标准模块的代码窗口中向模块中添加过程。

3. 类模块

类模块文件的扩展名为.cls。用户可以在类模块中编写代码建立新对象，这些新对象可以

包含自定义属性和方法，可以在应用程序内的过程中使用。其实，窗体本身就是这样一种类模块，在其上可放置控件、显示窗体窗口。

类模块与标准模块的区别为：标准模块仅含有代码，类模块既含有代码又含有数据。

## 9.2 变量的作用域和生存期

变量的作用域是指变量能被某一过程识别的范围，它决定了哪些子过程和函数过程可访问该变量。

### 9.2.1 变量的作用域

根据变量的作用域，变量可分为局部变量、窗体/模块级变量和全局变量。这 3 种变量的作用域及使用规则见表 9-1。

表 9-1 变量的作用域及使用规则

作用范围	局部变量	窗体/模块级变量	全局变量	
			窗体	标准模块
声明方式	Dim, Static	Dim, Private	Public	
声明位置	在过程中	窗体/模块的通用声明段	窗体/模块的通用声明段	
能否被本模块的其他过程存取	不能	能	能	
能否被其他模块存取	不能	不能	能，但在变量名前加窗体名	能

#### 1. 局部变量

局部变量是指在过程内用 Dim 或 Static 语句声明的变量，或不加声明直接使用的变量，它只能在本过程中使用，别的过程不可访问。例如：

```
Dim a As Single
Static x As String
```

局部变量随过程的调用而分配存储单元，并进行变量的初始化，在此过程体内进行数据的存取，一旦该过程体结束，变量的内容自动消失，占用的存储单元自动释放。

使用局部变量，在不同的过程中可以使用相同名称的变量，彼此互不干扰，这样便于程序的调试。

**【例 9-1】** 局部变量示例。根据窗体的事件代码和通用过程，分析输出结果。

窗体 Form 的 Activate 事件代码如下：

```
Private Sub Form_Activate()
    Dim a As Integer, b As Integer, c As Integer          ' 定义局部变量
    a = 4: b = 7
    Print
    Print Tab(31); "a"; Tab(41); "b"; Tab(51); "c=a+b"
    Print
    Print Tab(5); "调用子过程 test 前的变量值"; Tab(30); a; Tab(40); b; Tab(50); c
    Call test
```

Print Tab(5); "调用子过程 test 后的变量值"; Tab(30); a; Tab(40); b; Tab(50); c

End Sub

通用过程代码如下：

Sub test()

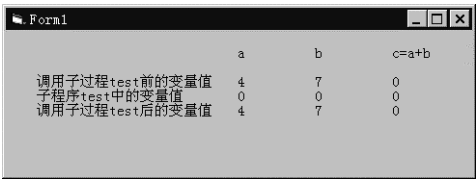
Dim a As Integer, b As Integer, c As Integer ' 定义局部变量

c = a + b

Print Tab(5); "子程序 test 中的变量值"; Tab(30); a; Tab(40); b; Tab(50); c

End Sub

运行程序，结果如图 9-5 所示，可以看出，主程序中的变量没有带到子过程中。



	a	b	c=a+b
调用子过程test前的变量值	4	7	0
子程序test中的变量值	0	0	0
调用子过程test后的变量值	4	7	0

图 9-5 局部变量的调用

一个较复杂的程序可能有多个过程或函数。在编写过程（函数）代码时，应该把注意力集中在这一相对独立的子过程内，其中所用到的变量如果都是局部变量，则无论怎样处理都不会影响到外界。但如果使用非局部变量，考虑不周时容易引起麻烦。所以，为安全起见，过程（函数）体内应尽可能用局部变量。

2. 窗体/模块级变量

窗体/模块级变量是指在一个窗体/模块的任何过程外，即在通用声明段中用 Private 或 Dim 语句声明的变量，可被本窗体/模块的任何过程中访问。例如：

Private x As String

Dim y As Integer

在模块的通用段中使用 Private 或 Dim 语句的作用相同，但使用 Private 语句会提高代码的可读性。

【例 9-2】 窗体/模块级变量示例。根据窗体的事件代码和通用过程，分析输出结果。在通用段中声明：

Private a As Integer, b As Integer, c As Integer ' 在通用段中定义变量

窗体 Form 的 Activate 事件代码如下：

Private Sub Form\_Activate()

a = 4: b = 7

Print

Print Tab(31); "a"; Tab(41); "b"; Tab(51); "c=a+b"

Print

Print Tab(5); "调用子过程 test 前的变量值"; Tab(30); a; Tab(40); b; Tab(50); c

Call test

Print Tab(5); "调用子过程 test 后的变量值"; Tab(30); a; Tab(40); b; Tab(50); c

End Sub

通用过程代码如下：

```
Sub prod()  
    c = a + b  
    Print Tab(5); "子程序 test 中的变量值"; Tab(30); a; Tab(40); b; Tab(50); c  
End Sub
```

运行程序，结果如图 9-6 所示。可以看出，窗体/模块级变量可被本模块的任何过程访问。

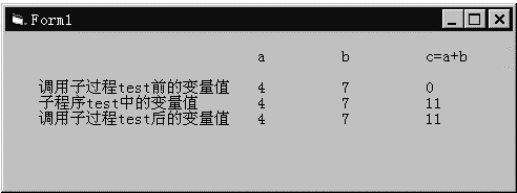


图 9-6 窗体/模块级变量的调用

3. 全局变量

全局变量是指只能在标准模块的任何过程或函数外，即在通用声明段中用 **Public** 语句声明的变量，可被应用程序的任何过程或函数访问。例如：

```
Public x As Single
```

全局变量的值在整个应用程序中始终不会消失和重新初始化，只有当整个应用程序执行结束时，才会消失。

在子过程中可以任意改变和调用全局变量，当子过程执行完后，其值又被带回主程序。把变量定义为全局变量虽然很方便，但有可能会使变量在程序中被无意修改。

【例 9-3】 变量使用方法示例。在一个标准模块文件中进行不同级的变量声明。

```
Public a As Integer           ' 全局变量  
Private b As String*10       ' 模块级变量  
Sub test1()  
    Dim x As Integer          ' 局部变量  
    .....  
End Sub  
Sub test2()  
    Dim y As String           ' 局部变量  
    .....  
End Sub
```

9.2.2 变量的生存期

变量的作用域是针对变量的作用空间而言的，而变量的生存期则是针对变量的作用时间来讲的。根据变量在程序运行期间的生命周期，变量可分为动态变量和静态变量。

1. 动态变量

动态（Dynamic）变量是指程序运行到变量所在的过程时，才分配该变量的内存单元，



经过处理退出该过程后，该变量占用的内存单元自动释放，其值消失，其内存单元能被其他变量使用。

使用 Dim 语句在过程中声明的局部变量属于动态变量，在过程执行结束后，变量的值不被保留。每执行一次过程，变量被重新声明一次。

## 2. 静态变量

静态 (Static) 变量是指程序运行到该变量所在的过程中，修改变量的值后，退出该过程，其值仍被保留，即变量所占的内存单元没有释放。当以后再次进入该过程时，原来变量的值可以继续使用。

使用 Static 语句在过程中声明的局部变量属于静态变量。

## 3. 动态变量与静态变量使用示例

【例 9-4】 运行下面的程序代码，分析运行结果。

窗体 Form 的 Activate 事件代码如下：

```
Private Sub Form_Activate()  
    Dim i As Integer  
    For i = 1 To 4                ' 循环 4 次  
        tests                    ' 调用通用过程 tests  
    Next i  
End Sub
```

通用过程代码如下：

```
Sub tests()  
    Dim a As Integer, x As String ' 定义动态变量 a 和 x，每次调用重新初始化  
    Static b, y                  ' 定义静态变量 b 和 y，保留变量的值  
    a = a + 1                    ' a 为动态变量  
    b = b + 1                    ' b 为静态变量  
    x = x & "*"                  ' x 为动态变量  
    y = y & "*"                  ' y 为静态变量  
    Print  
    Print Tab(5); "a="; a, " b="; b, "x="; x, "y="; y  
End Sub
```

运行程序，结果如图 9-7 所示。

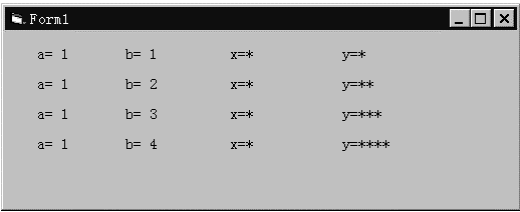


图 9-7 动态变量与静态变量示例

从运行情况可以看出，a, b, x, y 都是过程 tests 中的局部变量，b, y 被说明为静态变量，

每次调用都会保留上一次的值，b, y 的值会变化；a, x 是动态变量，每次调用都被重新初始化为 0 或" "，它们的值总是不变的。

如果需要将过程中的所有局部变量设置为静态变量，可在过程头的起始处加上 Static 关键字。例如：

Static Function a(n)

【例 9-5】 编写程序，要求利用文本框检查用户口令，使用静态变量来限制输入口令的次数。

【分析】假设提供给用户 3 次输入密码的机会，密码为“1234567890”。使用静态变量 n 来限制输入口令的次数，每输入一次密码，该变量累加一次。

设计步骤如下。

- （1）建立用户界面，如图 9-8 所示。
- （2）设置对象属性，见表 9-2。

表 9-2 属性设置

对 象	属 性	属 性 值	说 明
Text1	Text		清空
	PasswordChar	*	只显示字符“*”
Label1	Caption	请输入口令：	标题
Label2	Caption	您共有 3 次输入机会，好好把握！	标题

- （3）编写文本框的事件代码。

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
    Static n As Integer          ' 设置静态变量
    If KeyAscii = 13 Then        ' Enter 键
        If LCase(Text1.Text) = "1234567890" Then
            MsgBox "欢迎使用本系统", 0, "请多提宝贵意见"
        Else
            n = n + 1
            If n = 3 Then
                MsgBox "对不起，您无权使用本系统。", 0, "密码错误！"
                Text1.Enabled = False
            Else
                MsgBox "您还有" & Str(3 - n) & "次机会。", 0, "对不起，口令错！"
                Text1.SelStart = 0
                Text1.SelLength = Len(Text1.Text)
            End If
        End If
    End If
End Sub
```

运行程序，如果输入的密码正确，则显示“欢迎使用本系统”，如图 9-8 所示；如果输入

的密码不正确，则弹出消息框，提示您还有几次机会，如图9-9 和图9-10 所示；如果 3 次输入的密码都不正确，则显示“对不起，您无权使用本系统。”，如图 9-11 所示。

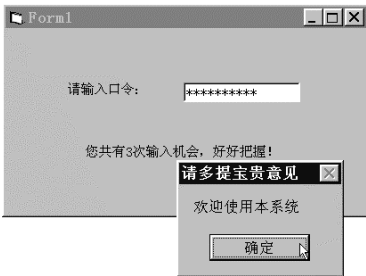


图 9-8 输入密码正确

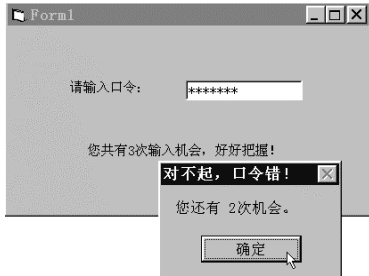


图 9-9 第一次输入错误

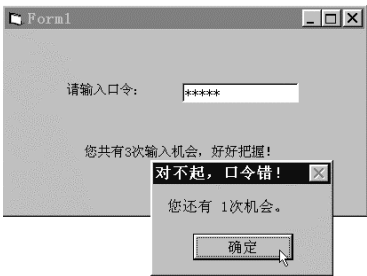


图 9-10 第二次输入错误

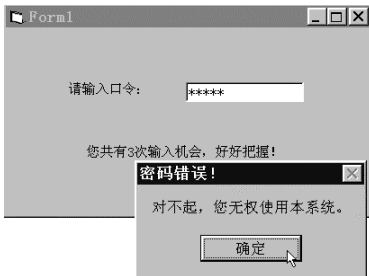


图 9-11 3 次输入的密码都错误

## 9.3 过程的作用域

VB 中过程的作用域分为模块级和全局级。

### 1. 模块级过程和全局级过程

#### （1）模块级过程

模块级过程是在某个模块内定义的过程，其作用域为本模块。

定义模块级过程的方法为：在 Sub 或 Function 前加关键字 Private，这样该过程只能被在本模块中定义的过程调用。

#### （2）全局级过程

全局级过程可被整个应用程序（工程）所有模块中定义的过程调用，其作用域为整个应用程序。

定义全局级过程的方法是：在 Sub 或 Function 前加关键字 Public（可以默认）。

### 2. 调用其他模块中的过程

在工程中的任何地方都能调用其他模块中的全局过程。调用其他模块中的过程的方法不同，取决于该过程是在窗体模块中、类模块中还是在标准模块中。

例如，所有窗体模块的外部调用必须指向包含此过程的窗体模块。如果在窗体模块 Form1 中包含 Sub1 过程，则可使用下面的语句调用 Form1 中的过程：

```
Call Form1.Sub1(x)
```

3. 过程的作用域及调用规则

过程的作用域及调用规则，参见表 9-3。

表 9-3 过程的作用域及调用规则

作用范围	模块级		全局级	
	窗体	标准模块	窗体	标准模块
定义方式	过程名前加 Private		过程名前加 Public 或默认	
能否被本模块其他过程调用	能	能	能	能
能否被本应用程序其他模块调用	不能	不能	能，但必须在过程名前加窗体名	能，但过程名必须唯一，否则要加标准模块名

9.4 按钮控件

通过前面的学习，我们已经知道命令按钮（CommandButton）控件是 Windows 应用程序常用的一种控件。当用户用鼠标单击或用 Enter 键按下命令按钮时，便可触发命令按钮的 Click 事件，从而执行其事件过程，达到完成某个特定操作的目的。本节将详细介绍按钮控件的一些常用属性的使用方法。

1. 命令按钮的属性

命令按钮具备控件所共有的一些基本属性，例如，Name、Height、Width、Top、Left、Font、Enabled 和 Visible 等，前面已做了介绍，这里不再赘述，下面介绍命令按钮的其他常用属性。

（1）Cancel 属性

该属性为逻辑型，系统默认值为 False。若设置为 True，则该按钮被定义为窗体的取消命令按钮，当用户按 Esc 键时，将自动激活该命令按钮的 Click 事件。

一个窗体只能有一个命令按钮被设置为取消命令按钮。若已有某命令按钮的 Cancel 属性被设置为 True，则其他命令按钮的 Cancel 属性将被系统自动设置为 False。

（2）Default 属性

该属性为逻辑型，与 Cancel 属性类似。若某命令按钮的 Default 属性被设置为 True，则该命令按钮就是窗体的默认命令按钮，当用户按 Enter 键时，该命令按钮的 Click 事件将自动激活。

一个窗体只能设置一个默认命令按钮。

（3）Value 属性

该属性为逻辑型，在设计阶段无效，可以在运行时设置或访问，用于检测命令按钮是否被按下。若被按下，则返回 True；否则，返回 False。

该属性的另一个重要用途是用于以程序方式来激活某命令按钮。当在程序中将某命令按钮的 Value 属性值设置为 True 或 1 时，将激活该命令按钮的 Click 事件，从而执行该事件过程。例如，要在程序中调用执行命令按钮 Command1 的 Click 事件过程，代码如下：

```
Command1.Value = 1
```

或

```
Command1.Value = True
```

(4) Style 属性

该属性为数值型，用于返回或设置命令按钮的外观风格。其取值为 0 或 1，分别代表标准 Windows 风格命令按钮和图形命令按钮。默认值为 0。该属性只能在属性窗口中设置。在图形命令按钮中，除可显示标题文字外，还可显示自定义的图形，使按钮图文并茂，更加形象直观。

(5) Picture 属性

该属性用于给命令按钮显示一幅示意图形，只有在 Style 属性设置为 1 时才有效。该属性可在属性窗口中直接设置，也可在运行时由程序代码设置。

(6) DisabledPicture 属性

该属性用于设置或返回当命令按钮失效时所显示的图形。该属性也只有在 Style 属性设置为 1 时才有效。要使命令按钮失效，只要将其 Enabled 属性设置为 False 即可。

2. 命令按钮的事件

命令按钮能响应绝大多数的事件，如 Click（单击），MouseMove（鼠标移动），DragDrop（拖放），KeyDown（键盘按下），KeyUp（键盘松开），KeyPress（按键），MouseDown（鼠标按下），MouseUp（鼠标松开）等事件，但最常用的是 Click 事件。

习题 9

一、选择题

- 9.1 以下叙述中错误的是（ ）。
- A) 在工程资源管理器窗口中只能包含一个工程文件及属于该工程的其他文件
  - B) 以.BAS 为扩展名的文件是标准模块文件
  - C) 窗体文件包含该窗体及其控件的属性
  - D) 一个工程中可以含有多个标准模块文件
- 9.2 如果一个工程含有多个窗体及标准模块，则以下叙述中错误的是（ ）。
- A) 如果工程中含有 Sub Main 过程，则程序一定首先执行该过程
  - B) 不能把标准模块设置为启动模块
  - C) 用 Hide 方法只是隐藏一个窗体，不能从内存中清除该窗体
  - D) 任何时刻最多只有一个窗体是活动窗体
- 9.3 以下关于变量作用域的叙述中，正确的是（ ）。
- A) 窗体中凡被声明为 Private 的变量只能在某个指定的过程中使用
  - B) 全局变量必须在标准模块中声明
  - C) 模块级变量只能用 Private 关键字声明
  - D) Static 类型变量的作用域是它所在的窗体或模块文件
- 9.4 下列叙述中正确的是（ ）。
- A) 在窗体的 Form\_Load 事件过程中定义的变量是全局变量

- B) 局部变量的作用域可以超出所定义的过程
- C) 在某个 Sub 过程中定义的局部变量可以与其他事件过程中定义的局部变量同名，但其作用域只限于该过程
- D) 在调用过程中，所有局部变量被系统初始化为 0 或空字符串

9.5 在窗体上画一个名称为 Command1 的命令按钮，然后编写如下事件过程：

```
Private Sub Command1_Click()
```

```
    Static x As Integer
```

```
    Cls
```

```
    For i=1 To 2
```

```
        y=y+x
```

```
        x=x+2
```

```
    Next
```

```
    Print x , y
```

```
End Sub
```

程序运行后，连续三次单击 Command1 按钮后，窗体上显示的是（ ）。

- A) 4    2
- B) 12    18
- C) 12    30
- D) 4    6

9.6 在窗体上画一个命令按钮，下列程序运行后，单击命令按钮，输出结果为（ ）。

```
Private Sub Command3_Click()
```

```
    Tcl 2
```

```
    Tcl 3
```

```
    Tcl 4
```

```
End Sub
```

```
Sub Tcl(a As Integer)
```

```
    Static x As Integer
```

```
    x=x + a
```

```
    Print x;
```

```
End Sub
```

- A) 2    3    4
- B) 2    5    9
- C) 3    5    4
- D) 2    4    3

9.7 设有如下通用过程：

```
Public Function f(x As Integer)
```

```
    Dim y As Integer
```

```
    x=20
```

```
    y=2
```

```
    f=x*y
```

```
End Function
```

在窗体上画一个名称为 Command1 的命令按钮，然后编写如下事件过程：

```
Private Sub Command1_Click()
```

```
    Static x As Integer
```

```
    x=10
```

```
y=5  
y=f(x)  
Print x ; y
```

**End Sub**

程序运行后，如果单击命令按钮，则在窗体上显示的内容是（ ）。

- A) 10 5                      B) 20 5                      C) 20 40                      D) 10 40

## 二、填空题

9.8 在窗体上画一个名称为 Command1 的命令按钮，然后编写如下事件过程：

**Private Sub Command1\_Click()**

```
Static y As Integer
```

```
Cls
```

```
For i=0 TO 2
```

```
    x=x+y
```

```
    y=y+3
```

```
Next
```

```
Print x , y
```

**End Sub**

程序运行后，连续单击两次 Command1 按钮后，窗体上显示的是\_\_\_\_\_。

## 三、编程题

9.9 编写程序，实现分数化简。要求：在标准模块中编写求最大公约数的函数过程，然后在窗体模块中调用它，来对分数进行化简。

9.10 利用全局变量和函数，设计模拟幸运数字机游戏。设幸运数字为 7，每次由计算机随机产生 3 个 0~10 之间的随机数，当这 3 个随机数中有一个数字为 7 时，就算赢了一次。要求：利用函数过程计算获胜率。

# 第 10 章 用户定义类型与枚举类型

前面介绍了 VB 的基本数据类型（整型、字符型、单精度型等），但是如果需要用 一个组合项来表示几个不同类型的数据时，用户可以将该组合项定义为自己的数据类型，即用户定义类型。

当某个变量的值只有几种可能时，如果需要将变量的值一一列举出来，可以将该变量定义为枚举类型。

## 10.1 用户定义类型

VB 中，系统提供了标准的数据类型来定义变量。但在实际工作中，常常遇到这样的问题：学生对象由学号、姓名、性别、分数等多项组成，即组合项，用这些组合项来描述相应对象的若干属性。这些描述相同对象的组合项的集合就形成了记录。

例如，某校的学生学籍管理情况见表 10-1。

表 10-1 学生学籍管理情况表

学 号	姓 名	性 别	政 治	语 文	数 学	平 均 分
2007001	张一光	女	70	80	90	80
2007002	李二琴	男	80	85	95	85
2007003	王三丽	男	90	92	88	90
2007004	赵四方	女	95	94	93	94
2007005	孙五成	男	90	95	85	90
2007006	关六强	男	76	82	93	84

在表10-1 中，表名“学生学籍管理情况表”对应磁盘中的数据文件，表中第一行中的 7 个栏目的集合就是记录，在它下面填入每个学生的 7 个具体属性值的集合就是记录值（简称记录），该表中共计有 6 个记录值，每位学生一条记录。

表中的每个栏目称为字段或数据项，字段的名称（如学号、姓名等）称为字段名或数据项名。

### 10.1.1 建立用户定义类型

记录与数组都是由多个数据项组成的。但是，二者最大的区别是：数组中的每个元素必须具有相同的数据类型，而记录中的每个数据项可以具有不同的数据类型。

#### 1. 语法规式

VB 提供了定义记录结构的语句，称这种语句为用户定义数据类型，其语法规式为：



〈字段名 1〉 As 〈类型名 1〉

● ● ● ● ● ●

## End Type

①〈用户类型名〉是用户定义的数据类型名，而不是变量，其命名规则与变量名的命名规则相同。

③ 〈类型名〉可以是任何基本数据类型，也可以是用户定义数据类型。

(1) 在使用用户定义类型之前，必须用 **Type** 语句定义。用户定义类型在标准模块中定义，其变量可以出现在工程的任何地方。

(3) 在用户定义类型中不能含有数组。

**【例 10-1】** 建立用户定义类型，使其包含学号、姓名、性别和平均分等数据项。用 Type 语句定义名称为 st 的记录，代码如下：

pf As Single ' 平均分为单精度型

## End Type

建立了用户定义类型后，就可以建立相应的用户定义类型变量，然后使用该变量。

建立用户定义类型后，可以用 `Dim`、`Redim` 或 `Static` 语句定义一个具有这种数据类型的变量。例如，在例 10-1 的基础上，定义一个具有 `st` 类型的变量 `stu`：

用户定义类型也可以作为数组元素的数据类型。例如，定义一个拥有 10 个记录元素的数组 `stud`:

如果要对用户定义类型变量中的某个字段的数据进行读取，使用方法为：

〈用户定义类型变量名〉.〈字段名〉

例如，要读取用户定义类型变量 stu 中 xh 字段的数据，要写为：stu.xh。

3. 建立和使用用户定义类型变量示例

【例 10-2】 如图 10-1 所示，利用用户定义类型变量对各项赋值。  
设计步骤如下。

(1) 建立用户界面并设置对象属性，如图 10-1 所示。

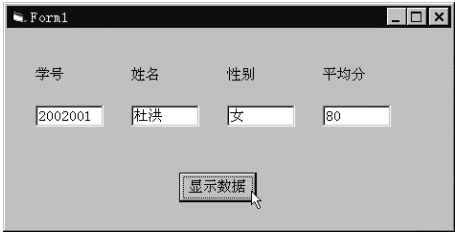


图 10-1 利用用户定义类型变量对各项进行赋值

(2) 在窗体模块的通用段创建用户定义类型：

```
Private Type st
    xh As String * 7           ' 学号定义为 7 个字符的定长字符串
    xm As String * 8           ' 姓名定义为 8 个字符的定长字符串
    xb As String * 2           ' 性别定义为 2 个字符的定长字符串
    pjf As Single               ' 平均分定义为单精度型
End Type
```

编写“显示数据”命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()
    Dim stu As st               ' 定义 st 类型的变量 stu
    stu.xh = "2002001"
    stu.xm = "杜洪"
    stu.xb = "女"
    stu.pjf = 80
    Text1.Text = stu.xh         ' Text1.Text 赋值为 stu.xh 的值
    Text2.Text = stu.xm         ' Text2.Text 赋值为 stu.xm 的值
    Text3.Text = stu.xb         ' Text3.Text 赋值为 stu.xb 的值
    Text4.Text = stu.pjf        ' Text4.Text 赋值为 stu.pjf 的值
End Sub
```

运行程序，结果如图 10-1 所示。

10.1.3 用户定义类型数组

所谓用户定义类型数组，是指其数组元素的数据类型为 用户定义类型，这种数组也称为记录数组（Array of records）。

1. 存取用户定义类型数组元素

存取用户定义类型数组元素的某字段数据的语法格式为：

〈用户定义类型数组元素〉.〈字段名〉

例如，要存取例 10-2 中第 2 个、第 3 个学生的平均分，应写成下面的形式：

```
stud(2).pjf
stud(3).pjf
```

2. 用户定义类型数组示例

**【例 10-3】** 定义一个包含 40 个学生信息的数组，试对第 3 个学生的学号、姓名、性别、平均分字段进行赋值。

设计步骤如下。

- (1) 建立用户界面并设置对象属性，如图 10-2 所示。
- (2) 在窗体模块的通用段创建用户定义类型。

Private Type st

```
xh As String * 7           ' xh 定义为 7 个字符的定长字符串
xm As String * 8           ' xm 定义为 8 个字符的定长字符串
xb As String * 2           ' xb 定义为 2 个字符的定长字符串
pjf As Single              ' pjf 定义为单精度型
```

End Type

- (3) 编写“显示数据”命令按钮 Command1 的 Click 事件代码。

Private Sub Command1\_Click()

```
Dim stud(1 To 40) As st      ' 定义用户定义数组
stud(3).xh = "2002010"      ' 为数组元素的 xh 字段赋值
stud(3).xm = "马飞飞"        ' 为数组元素的 xm 字段赋值
stud(3).xb = "女"            ' 为数组元素的 xb 字段赋值
stud(3).pjf = 94              ' 为数组元素的 pjf 字段赋值

Text1.Text = stud(3).xh
Text2.Text = stud(3).xm
Text3.Text = stud(3).xb
Text4.Text = stud(3).pjf
```

End Sub

运行程序，结果如图 10-2 所示。

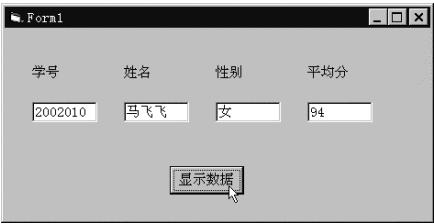


图 10-2 用户定义类型数组示例

# 10.2 枚举类型

当一个变量只有几种可能的值时，可以定义为枚举类型。所谓“枚举”，是指将变量的值一一列举出来，且变量的值只限于列举出来的值的范围。这样，可以方便地处理有关的常数，或者使名称与常数值相关联。例如，可以把与“星期”相关联的一组整数常数声明为一个枚举类型，然后在代码中使用“星期”的名称而不使用其整数数值。

## 10.2.1 定义枚举类型

枚举类型放在窗体模块、标准模块或公用模块中的声明部分，通过 Enum 语句来定义。

### 1. 定义枚举类型的语法格式

定义枚举类型的语法格式为：

```
[ Public | Private ] Enum <类型名称>
    <成员名> [= <常数表达式> ]
    <成员名> [= <常数表达式> ]
    .....
End Enum
```

【说明】

- ① 系统默认为 Public，表示所定义的枚举类型在整个工程中都是可见的。
- ② 选用 Private 关键字时，表示所定义的枚举类型只在声明的模块中可见。
- ③ <类型名称> 是指所定义的枚举类型的名称，必须是一个合法的 VB 标识符，在定义 Enum 类型的变量或参数时用该名称来指定类型。
- ④ <成员名> 必须是合法的 VB 标识符，用来指定所定义的枚举类型组成元素的名称。

### 2. 枚举类型中的常数值

在定义枚举类型的语法格式中，<常数表达式> 是可选项。

(1) 在默认情况下，枚举中的第一个常数被初始化为 0，其后的常数则被初始化为比其前面的常数大 1 的数值。例如：

```
Public Enum Days
    Sunday
    Monday
    Tuesday
    Wednesday
    Thursday
    Friday
    Saturday
End Enum
```

上面定义了一个枚举类型 Days，它包括 7 个成员。由于省略了 <常数表达式>，因此常数 Sunday 的值为 0，常数 Monday 的值为 1，常数 Tuesday 的值为 2，……

(2) 可以用赋值语句显式地给枚举中的常数赋值，即不省略 <常数表达式>，所赋的值

可以是任何长整数，包括负数。

如果想用小于 0 的常数表示出错条件，可以给枚举常数赋一个负值。

例如，在下面的枚举中，Saturday 是枚举中的第一个元素，被赋值为 0。由于 Sunday 被显式地赋为 0，则 Monday 被赋值为 1，Tuesday 为 2，其余类推。

```
Public Enum WorkDays
    Saturday
    Sunday=0
    Monday
    Tuesday
    Wednesday
    Thursday
    Friday
    Invalid = -1
```

End Enum

(3) 当对一个枚举中的常数赋值时，还可以使用另一个枚举中的常数值。但应注意，为避免混淆，应在常数名前加上枚举名。例如：

```
Public Enum WorkDays
    Sunday=0
    Monday
    Tuesday
    Wednesday
    Thursday
    Friday
    Saturday=Days.Saturday-6
    Invalid=-1
```

End Enum

(4) VB 将枚举中的常数数值看做是长整数。如果将一个浮点数值赋给一个枚举中的常数，VB 会自动将该值取整为最接近的长整数。

### 10.2.2 枚举类型使用示例

【例 10-4】 输入一个表示星期的数字，判断该天是工作日还是非工作日。

【分析】先定义枚举类型 Workdays，再定义一个 Workdays 类型的变量 Myday。Myday 的取值范围为 0~6。当 Myday 的值是 6（表示星期六）或 0（表示星期日）时，输出是“非工作日”，否则输出“工作日”。

设计步骤如下。

- (1) 建立用户界面并设置对象属性，如图 10-3 所示。
- (2) 编写事件代码。

建立新工程，执行“工程”→“添加模块”菜单命令，在代码窗口中输入下面的枚举类型定义：

**Public Enum Workdays**

Sunday	' 值为 0
Monday	' 值为 1
Tuesday	' 值为 2
Wednesday	' 值为 3
Thursday	' 值为 4
Friday	' 值为 5
Saturday	' 值为 6
Invalid = -1	

**End Enum**

编写“判断”命令按钮 Command1 的 Click 事件过程：

**Private Sub Command1\_Click()**

```
Dim Myday As Workdays          ' 定义 Myday 为枚举类型 Workdays
Text1.SetFocus
Myday = Val(Text1.Text)
If Myday = Sunday Or Myday=Saturday Then      ' 星期日或星期六
    Label4.Caption = "非工作日！可以休息了。^o^"
Else
    Label4.Caption = "工作日！不要偷懒呀。~！"
End If
End Sub
```

**End Sub**

运行程序，在文本框中输入的数字，结果如图 10-3 所示。

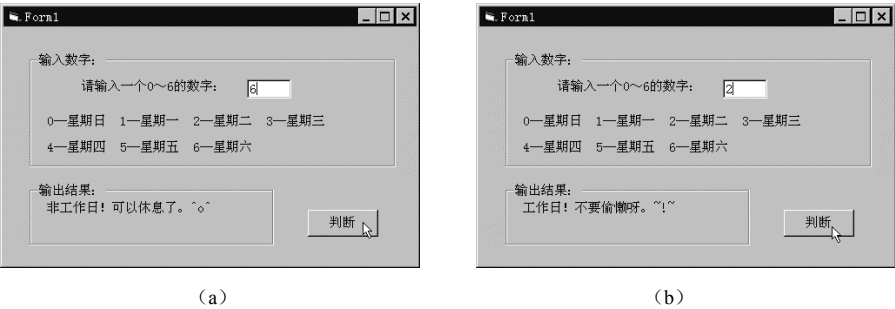


图 10-3 判断是否为工作日

当在代码窗口中输入示例中的代码时，VB 在自动列出成员列表中显示 Workdays 枚举的常数，如图 10-4 所示。

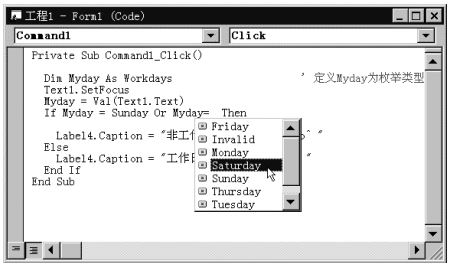


图 10-4 VB 自动显示 Workdays 枚举的常数

## 10.3 滚动条控件

滚动条（ScrollBar）的作用有两个：第一，附在窗体上协助观察数据或确定位置；第二，作为数据输入的工具。

当应用程序或控件所包含的信息超过当前窗口所能显示的信息时，滚动条就会自动出现。如果文本框或 MDI 窗体的 ScrollBars 属性设置为 True，则滚动条也会自动出现。

### 10.3.1 滚动条控件的类型

VB 的滚动条控件不同于 Windows 内部的滚动条，也不同于 VB 中附加在文本框、列表框、组合框或 MDI 窗体上的滚动条。滚动条控件为不能自动支持滚动的应用程序和控件提供了滚动功能。另外，还可以用滚动条作为输入设备。

滚动条有水平和垂直两种，可以通过水平滚动条（HScrollBar）和垂直滚动条（VScrollBar）工具来建立，如图 10-5 所示。除了方向不同之外，水平滚动条和垂直滚动条的动作是相同的。

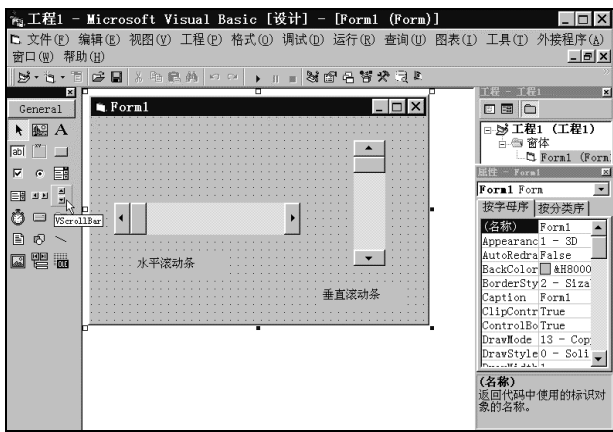


图 10-5 水平滚动条和垂直滚动条

#### 1. 水平滚动条

水平滚动条的滑块在最左端时代表最小值 Min，滑块由左往右移动，代表的值随之递增，在最右端代表最大值 Max。

#### 2. 垂直滚动条

垂直滚动条的滑块在最上端时代表最小值 Min，滑块由上向下移动，代表的值随之递增，到最下端为最大值 Max。

### 10.3.2 滚动条控件的常用属性

#### 1. Min 和 Max 属性

Min 和 Max 属性用来返回或设置滚动条所能代表的最小值和最大值，其取值范围为：-32768~32767。

Min 属性的默认值为 0，Max 属性的默认值为 32767。

## 2. Value 属性

Value 属性用来返回或设置滚动条的当前位置，其返回值始终介于 Max 和 Min 属性值之间，包括这两个值。

## 3. LargeChange 属性

LargeChange 属性用来返回或设置单击滑块和滚动箭头之间的区域时（称为滑杆），滚动条控件 Value 属性值的改变量。例如，设置 LargeChange 属性值为 20，若单击水平滑块左边的区域，则滚动条的 Value 属性值将减小 20；若单击滑块右边的区域，则 Value 值将增大 20。该属性的默认值为 1。

## 4. SmallChange 属性

SmallChange 属性用来返回和设置当用户单击滚动箭头时，滚动条控件 Value 属性值的改变量。当单击滚动条两端的箭头按钮时，滚动条的值将按最小改变量进行递增或递减。该属性的默认值为 1。

### 10.3.3 滚动条控件的常用事件

滚动条控件可以识别的事件中，较重要的是 Change 事件和 Scroll 事件。

#### 1. Change 事件

在程序运行过程中，每当滚动条的 Value 属性发生变化时，就发生 Change 事件。而每当用户单击滚动箭头、单击滑块与滚动箭头之间的区域或沿着滚动条拖拉滑块的动作结束时，滚动条的 Value 属性就发生变化。

在实际应用中，由于在单击滚动条或滚动箭头时，将产生 Change 事件，因此常利用 Change 事件来获得滚动条变化后的最终值。

#### 2. Scroll 事件

尽管拖动滑块会引起 Value 属性发生变化，从而触发 Change 事件，但在滚动条内拖动滑块的过程中，并不发生 Change 事件，此时将触发产生滚动条的 Scroll（滚动）事件。当然，滑块的位置改变后，又将触发产生 Change 事件。

在实际编程中，常用 Scroll 事件过程来跟踪滚动条在拖动时数值的动态变化。

### 10.3.4 滚动条控件使用示例

使用滚动条可以较方便地实现定位，通常还利用滚动条作为输入设备，以及作为颜色、速度、数量的指示器。

**【例 10-5】** 滚动条控件的简单示例。

在窗体上建立一个水平滚动条，一个文本框（用来显示滑块当前位置所代表的值），如图 10-6 所示。



为了让文本框能够即时显示滑块当前位置所代表的值，必须要有下面的事件过程代码：

```
Private Sub HScroll1_Change()  
    HScroll1.Max = 200  
    HScroll1.Min = 1  
    HScroll1.SmallChange = 1  
    HScroll1.LargeChange = 5  
    Text1.Text = HScroll1.Value  
  
End Sub
```

运行程序，结果如图 10-7 所示。

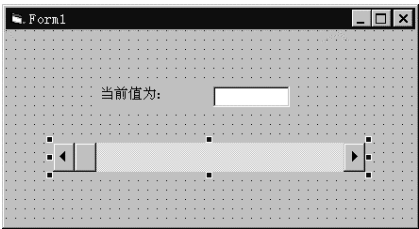


图 10-6 建立界面

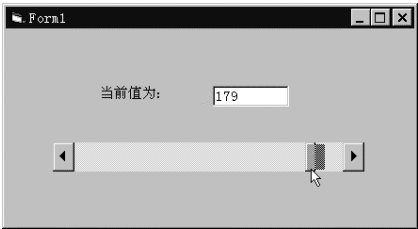


图 10-7 水平滚动条简单示例

**【例 10-6】** 设计一个通过滚动条调整颜色的程序。程序启动后的界面如图 10-8 所示，当用户单击 3 个滚动条两端的箭头按钮、直接拖动滚动条上的滑块或单击滚动条上的滑杆时，可以调整 `Rgb()` 函数中对应的颜色值，从而使“颜色效果”区中显示出不同的颜色。

设计步骤如下。

(1) 建立用户界面。

在窗体中添加 3 个水平滚动条 `HScroll1~HScroll3`，3 个标签 `Label1~Label3`，一个框架 `Frame1`，并在框架中添加一个文本框 `Text1`。调整控件大小并安排适当的位置，如图 10-9 所示。

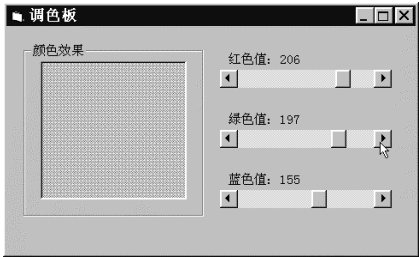


图 10-8 滚动条的应用

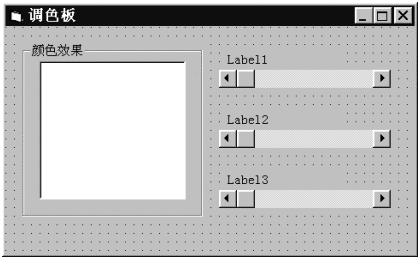


图 10-9 用户界面

(2) 设置对象属性。

将窗体的 `Caption` 属性设为“调色板”。

将 `Label1~Label3` 的 `AutoSize` 属性设为 `True`。

将 `HScroll1~HScroll3` 的 `Max` 属性设为 255，`SmallChange` 属性设为 8，`LargeChange` 属性设为 32。

将 `Frame1` 的 `Caption` 属性设为“颜色效果”。

将 `Text1` 的 `Text` 属性设为空，`Enabled` 属性设为 `False`（不响应用户产生的事件）。

(3) 编写事件代码。

创建 ChangeColor 过程如下：

```
Private Sub ChangeColor()  
    Text1.BackColor = RGB(HScroll1.Value, HScroll2.Value, HScroll3.Value)  
  
End Sub
```

编写窗体 Form1 的 Load 事件代码：

```
Private Sub Form_Load()  
    Label1.Caption = "红色值：" & HScroll1.Value  
    Label2.Caption = "绿色值：" & HScroll2.Value  
    Label3.Caption = "蓝色值：" & HScroll3.Value  
    ChangeColor                ' 调用 ChangeColor 过程  
  
End Sub
```

编写水平滚动条 HScroll1 的 Change（改变）事件代码：

```
Private Sub HScroll1_Change()  
    Label1.Caption = "红色值：" & HScroll1.Value  
    ChangeColor                ' 调用 ChangeColor 过程  
  
End Sub
```

编写水平滚动条 HScroll1 的 Scroll（滚动）事件代码：

```
Private Sub  
    Label1.Caption = "红色值：" & HScroll1.Value  
    ChangeColor                ' 调用 ChangeColor 过程  
  
End Sub
```

编写水平滚动条 HScroll2 的 Change 事件代码：

```
Private Sub HScroll2_Change()  
    Label2.Caption = "绿色值：" & HScroll2.Value  
    ChangeColor                ' 调用 ChangeColor 过程  
  
End Sub
```

编写水平滚动条 HScroll2 的 Scroll 事件代码：

```
Private Sub HScroll2_Scroll()  
    Label2.Caption = "绿色值：" & HScroll2.Value  
    ChangeColor                ' 调用 ChangeColor 过程  
  
End Sub
```

编写水平滚动条 HScroll3 的 Change 事件代码：

```
Private Sub HScroll3_Change()  
    Label3.Caption = "蓝色值：" & HScroll3.Value  
    ChangeColor                ' 调用 ChangeColor 过程  
  
End Sub
```

编写水平滚动条 HScroll3 的 Scroll 事件代码：

```
Private Sub HScroll3_Scroll()
```

```
Label3.Caption = "蓝色值: " & HScroll3.Value  
ChangeColor          ' 调用 ChangeColor 过程
```

**End Sub**

运行程序，结果如图 10-8 所示。

## 习题 10

### 一、选择题

10.1 假定在窗体(名称为 Form1)的代码窗口中定义如下记录类型:

**Private Type animal**

```
animalName As String *20
```

```
aColor As String *10
```

**End Type**

在窗体上画一个名称为 Command1 的命令按钮，然后编写如下事件过程:

**Private Sub Command1\_Click()**

```
Dim rec As animal
```

```
Open"c:\vbTest.dat" For Random As#1 Len=Len(rec)
```

```
rec.animalName="Cat"
```

```
rec.aColor="White"
```

```
Put#1 , , rec
```

```
Close #1
```

**End Sub**

则以下叙述中正确的是 ( )。

- A) 记录类型 animal 不能在 Form1 中定义，必须在标准模块中定义
- B) 如果文件 c:\vbTest.dat 不存在，则 Open 命令执行失败
- C) 由于 Put 命令中没有指明记录号，因此每次都把记录写到文件的末尾
- D) 语句“Put #1 , , rec”将 animal 类型的两个数据元素写到文件中

10.2 表示滚动条控件取值范围最大值的属性是 ( )。

- A) Max
- B) LargeChange
- C) Value
- D) Max-Min

10.3 当在滚动条内拖动滚动块时触发 ( )。

- A) KeyUp 事件
- B) KeyPress 事件
- C) Scroll 事件
- D) Change 事件

### 二、填空题

10.4 Visual Basic 对象可以分为两类，分别为\_\_\_\_和\_\_\_\_。

10.5 当滚动条位于最左端或最上端时，Value 属性被设置为\_\_\_\_。

### 三、编程题

10.6 输入学生的姓名、学号、语文成绩、英语成绩、数学成绩，计算每名学生的平均成绩，并显示各科成绩，如图 10-10 所示。

输入记录中各数据

学号

姓名

语文

外语

数学

学号 姓名 语文 外语 数学 平均成绩

List1

输入

删除

结束

(a)

输入记录中各数据

学号

姓名

语文

外语

数学

学号 姓名 语文 外语 数学 平均成绩

20004	丁松	80	90	95	88
20005	刘程	75	94	93	87
20006	刘宏	80	84	92	85
20007	赵萍	85	89	84	86
20008	孙岳	93	92	90	91
20009	殷梅	89	81	86	85
20010	李顶	90	98	96	94
20011	李原	80	92	96	89
20012	刘宏	80	82	85	82

输入

删除

结束

(b)

图 10-10 输入学生情况，计算平均分、显示各科成绩

- 10.7 为习题 10.6 增加计算全班平均成绩的功能。
- 10.8 为习题 10.6 增加按某科成绩排序的功能。
- 10.9 为习题 10.6 增加按姓名查找的功能。

# 第 11 章 图形与图像

VB 为用户提供了简捷有效的图形图像处理能力。除了窗体和控件的图形图像特征以外，它还提供了一系列基本的图形函数、语句和方法，支持直接在窗体上产生图形、图像和颜色，控制对象的位置和外观等。

## 11.1 绘制图形

VB 提供了两种绘图方式：第一是使用绘图控件，如 Line 控件、Shape 控件；第二是使用绘图方法，如 Line 方法、Circle 方法等。使用绘图控件无须编写代码，但它提供的绘图样式选择有限，只能实现简单功能，要实现更高级功能，还得采用绘图方法。

### 11.1.1 图形控件

Shape（形状）控件和 Line（直线）控件可用在窗体表面画图形元素。这些控件不支持任何事件，只用于表面装饰。可以在设计时通过设置其属性来确定显示的图形，也可以在程序运行时修改属性以动态地显示图形。

#### 1. Shape 控件

Shape 控件预定义了 6 种形状的图形，可通过设置 Shape 属性来显示所需的图形，见表 11-1。

表 11-1 Shape 属性设置值

属 性 值	常 数	说 明
0	VbShapeRectangle	（默认值）矩形
1	VbShapeSquare	正方形
2	VbShapeOval	椭圆形
3	VbShapeOval	圆形
4	VbShapeRoundedRectangle	圆角矩形
5	VbShapeRoundedSquare	圆角正方形

可以调整这些形状的大小，设置其颜色、边框样式、边框宽度等。表 11-2 列出了 Shape 控件的常用属性。

表 11-2 Shape 控件的常用属性

属 性	说 明	属 性	说 明
BorderColor	边框色	BorderWidth	边框宽度
FillColor	填充色	FillStyle	填充样式
BorderStyle	边框样式	DrawMode	画图模式

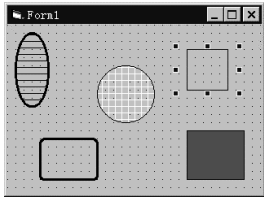


图 11-1 设计时用 Shape 控件画的图形

BorderStyle 属性产生的效果取决于 BorderWidth 属性的设置。如果 BorderWidth 值不是 1, 并且 BorderStyle 值不是 0 或 6, 则将 BorderStyle 值设置成 1。图 11-1 为设计时用 Shape 控件画的各种图形。

可以在容器中绘制 Shape 控件, 但是不能把 Shape 控件当做容器。

## 2. Line 控件

Line 控件与 Shape 控件相似, 但仅用于画线。Line 控件用于在窗体、图片框或框架中画各种直线段, 既可以在设计时通过设置线的端点坐标属性来画出直线, 也可以在程序运行时动态地改变直线的各种属性。

在设计时, 可以使用 Line 控件在窗体上可视化地安排直线的位置、长度、颜色、宽度、实虚线等属性。运行时不能使用 Move 方法移动 Line 控件, 但是可以通过改变 X1, X2, Y1 和 Y2 属性来移动它或者调整它的大小。

BorderStyle 属性的效果取决于 BorderWidth 属性的设置。如果 BorderWidth 值不是 1, 并且 BorderStyle 值不是 0 或 6, 则将 BorderStyle 值自动设置成 1。可以在运行时修改其属性, 语句如下:

```
Line.BorderWidth=3 ' 将直线宽度设置为 3 个像素 (Pixel)
```

用 Line 控件画直线, 系统将每一根直线 (即一个 Line 控件) 都看成是一个对象。

## 3. 图形控件与动画

**【例 11-1】** 曲柄滑块机构的演示。利用 Timer 控件来控制图形控件的转动, 如图 11-2 所示。

设计步骤如下。

(1) 建立应用程序用户界面。新建一个工程, 进入窗体设计器。首先增加一个框架 Frame1 和一个命令按钮 Command1。选定 Frame1, 在其中增加 4 个形状控件 Shape1 (大圆), Shape2 (滑块), Shape3 (圆周上的动点) 和 Shape4 (圆心), 一个计时器控件 Timer1 和若干直线控件 Line1 (连杆)、Line2 (半径) 等, 如图 11-3 所示。

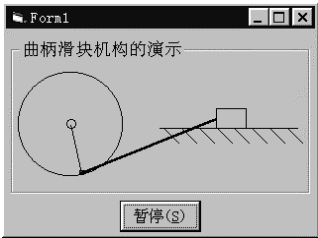


图 11-2 曲柄滑块机构

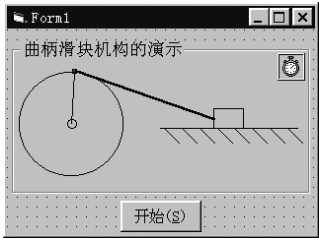


图 11-3 用户界面的设计

其中, 计时器控件 Timer1 可以放在窗体的任何位置。

(2) 设置对象属性, 参见表 11-3。其他属性的设置, 参见图 11-3。

表 11-3 属性设置

对 象	属 性	属 性 值	说 明
Shape1	Shape	3 - Circle	圆形
Shape3	Shape	3 - Circle	圆形
	FillColor	(红色)	填充颜色
	FillStyle	0 - Solid	填充类型
Shape4	Shape	3 - Circle	圆形
Line1	BorderWidth	2	连杆的宽度
Timer1	Interval	100	
	Enabled	False	

(3) 编写程序代码。

在通用模块中声明符号常数及窗体级变量：

```
Const pi = 3.14159
```

```
Dim X0 As Single, Y0 As Single, t As Integer
```

编写窗体的 Load 事件代码：

```
Private Sub Form_Load()
```

```
With Shape1
```

```
    .Tag = .Width / 2                ' 圆的半径
```

```
    X0 = .Left + .Tag                ' 圆心的 x 坐标
```

```
    Y0 = .Top + .Tag                ' 圆心的 y 坐标
```

```
End With
```

```
With Line1                          ' 连杆的长
```

```
    .Tag = Sqr((.X1 - .X2) ^ 2 + (.Y1 - .Y2) ^ 2)
```

```
End With
```

```
End Sub
```

编写 Timer1 的 Timer 事件代码：

```
Private Sub Timer1_Timer()
```

```
t = t + 1
```

```
Shape3.Left = X0 + Shape1.Tag * Sin(pi * t / 30) - 30
```

```
Shape3.Top = Y0 - Shape1.Tag * Cos(pi * t / 30) + 30
```

```
Line1.X1 = Shape3.Left + 30
```

```
Line1.Y1 = Shape3.Top + 30
```

```
Line1.X2 = Shape3.Left + Sqr(Line1.Tag ^ 2 - (Shape3.Top - Y0) ^ 2)
```

```
Line2.X1 = Line1.X1
```

```
Line2.Y1 = Line1.Y1
```

```
Shape2.Left = Line1.X2
```

```
End Sub
```

编写命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()
```

```
If Command1.Caption = "暂停(&S)" Then
    Command1.Caption = "继续(&C)"
    Timer1.Enabled = False
Else
    Command1.Caption = "暂停(&S)"
    Timer1.Enabled = True
End If
End Sub
```

11.1.2 图形的坐标系统

每一个图形操作（包括调整大小、移动和绘图），都要使用绘图区或容器的坐标系统。坐标系统是一个二维网格，可定义屏幕上、窗体中或其他容器中（如图片框或 Printer 对象）的位置。

任何容器的默认坐标系统，都是由容器的左上角（0,0）坐标开始的。沿坐标轴定义位置的测量单位，统称为刻度。在 VB 中，坐标系统的每个轴都有自己的刻度。

窗体的坐标系统，如图 11-4 所示。

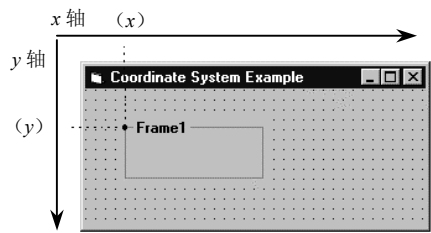


图 11-4 窗体的坐标系统

坐标轴的方向、起点和坐标系统的刻度，都是可以改变的，一般使用的是默认系统。

1. 坐标单位

坐标单位就是坐标的刻度。默认的坐标系统采用 twip 为单位。设置对象的 ScaleMode 属性可以改变坐标系统的单位，例如，可以采用像素或毫米为单位。表 11-4 列出了 ScaleMode 属性可能的取值。

表 11-4 ScaleMode 属性设置值

属 性 值	常 数	说 明
0	VbUser	自定义坐标系统
1	VbTwips	twip（默认值，1440 twip/inch，567 twips/cm）
2	VbPoints	磅（72 磅/inch）
3	VbPixels	pixel（显示器分辨率的最小单位）
4	VbCharacters	字符（水平每个单位=120 twip，垂直每个单位=240 twip）
5	VbInches	inch
6	VbMillimeters	mm
7	VbCentimeters	cm



例如，下面的代码使窗体的坐标单位改为毫米：

```
ScaleMode = 6
```

2. 坐标方法

使用 Scale 方法也可以设置用户的坐标系统，其语法格式为：

```
[〈对象〉.] Scale(x1,y1) - (x2,y2)
```

【说明】(x1,y1)设置〈对象〉的左上角坐标，(x2,y2)设置〈对象〉的右下角坐标。使用 Scale 方法将把〈对象〉在 x 方向上分为 x2-x1 等份，在 y 方向上分为 y2-y1 等份。使用 Scale 方法将自动把 ScaleMode 属性设置为 0。

3. 坐标属性

与坐标系统有关的属性，见表 11-5。

表 11-5 坐标属性

属 性	说 明
ScaleTop	对象左上角的纵坐标
ScaleLeft	对象左上角的横坐标
ScaleWidth	对象右下角的横坐标
ScaleHeight	对象右下角的纵坐标
CurrentX	当前点的横坐标
CurrentY	当前点的纵坐标

例如，代码：

```
Me.ScaleTop = 10
Me.ScaleLeft = 10
Me.ScaleWidth = 90
Me.ScaleHeight = 110
```

与代码：

```
Me.Scale(10,10) - (110,90)
```

是等效的。

11.1.3 与图形有关的属性

在 VB 中，窗体、图片框或 Printer 对象都有一些与图形有关的属性，使用这些属性能够设置颜色、线型和填充样式，并绘制出丰富多彩的图形。表 11-6 中列出了这些图形属性。

表 11-6 对象的图形属性

类 别	属 性
显示处理	AutoRedraw, ClipControls
绘图技术	DrawMode, DrawStyle, DrawWidth, BorderStyle, BorderWidth
填充技术	FillColor, FillStyle
颜色	BackColor, ForeColor, BorderColor, FillColor

### 1. DrawMode 属性

DrawMode 属性用来决定由图形方法或 Shape 控件及 Line 控件所绘制线条的真实颜色。当然，在一个黑色或纯白色的背景上，或者在未定义颜色的背景上绘图时，DrawMode 属性将不起作用。

用 DrawMode 属性绘图时，系统将当前的 ForeColor、BackColor 和 DrawMode 属性所确定的方式组合起来，其结果为最后绘图的颜色。DrawMode 属性的语法为：

[ <对象> .] DrawMode [= <值> ]

其中，<值> 的范围为 1~16，按其复杂程度可分为表 11-7、表 11-8、表 11-9 和表 11-10 中列出的 4 类。

表 11-7 最简单的 DrawMode 功能

值	功 能
1	像素变黑
16	像素变白
11	像素颜色保持原色不变，即空操作（NOP），效果相当于关闭绘图
6	像素变成其补色，即所有颜色数据变反（0 变 1，1 变 0）

在这 4 种变化中，DrawMode 属性不考虑 ForeColor，只设置像素的当前颜色，具体功能见表 11-7。

表 11-8 与 ForeColor 有关的 DrawMode 功能

值	功 能
13（CopyPen）	像素变成前景色 ForeColor，不管原来是什么颜色。这是默认的 DrawMode 值，也是最常见的设置值
4（NotCopyPen）	像素变成前景色 ForeColor 的补色

在这两种变化中，像素颜色只与 ForeColor 属性有关，而与 BackColor 属性及当前颜色无关，具体功能见表 11-8。

表 11-9 与 Xor（异或）有关的 DrawMode 功能

值	功 能
7（XorPen）	将当前颜色与 ForeColor 进行异或（Xor）操作
10（NotPen）	将当前颜色与 ForeColor 进行异或操作，再将操作的结果进行非（Not）操作

DrawMode 属性的这种功能与 ForeColor 和当前颜色有关，具体功能见表 11-9。在绘图和动画中，DrawMode=7 是一个很有用的设置值，它利用 Xor 方式，在同一地点绘图两次，从而消除第一次绘画的痕迹，精确地还原出绘图之前的显示内容，从而实现动画。

任何颜色与其自身进行 Xor 操作将产生黑色，进行 NotXor 操作将产生白色，见表 11-10。

表 11-10 DrawMode 的合并操作功能

值	功 能
15（MergePen）	将当前颜色与笔颜色进行或（Or）操作
2（NotMergePen）	将当前颜色与笔颜色进行或（Or）操作后，再对结果进行非（Not）操作
12（MergeNotPen）	先对笔颜色进行非（Not）操作，然后再与当前颜色进行或（Or）组合
14（MergeNotPen）	先对当前颜色进行非（Not）操作，再与笔颜色进行或（Or）组合

由上表可以看出，这些组合的结果是难以预测的。

使用 Mask 类操作，可以为 CopyPen 建立图像的阴影，见表 11-11。

在 DrawMode 的 16 种设置中，只有值 1, 6, 7, 11, 13, 16 的结果是能预测的。

表 11-11 DrawMode 的 Mask 类操作功能

值	功 能
9 (MaskPen)	将笔颜色与当前颜色进行 And 操作
8 (NotMaskPen)	将笔颜色与当前颜色进行 And 操作，然后再对结果进行 Not 操作
3 (MaskNotPen)	先对笔颜色进行 Not 操作，再将结果与笔颜色进行 And 操作
5 (MaskPenNot)	先对当前颜色进行 Not 操作，再将结果与笔颜色进行 And 操作

2. DrawWidth 属性和 DrawStyle 属性

(1) DrawWidth 属性

其语法格式为：

```
[ <对象> .] DrawWidth [= <值> ]
```

窗体、图片框的 DrawWidth 属性可以用来设置绘图线的宽度，<值>以像素为单位，设置后影响 PSet, Line 和 Circle 方法。<值>的范围是 1~32 767，默认值为 1，也就是说，画出的线为 1 个像素宽。

(2) DrawStyle 属性

其语法格式为：

```
[ <对象> .] DrawStyle [= <值> ]
```

DrawStyle 属性用于指定用图形方式创建的线是实线还是虚线。<值>的取值范围为 0~6，用来产生不同间隔的实、虚线，默认值为 0（实线）。

不同设置值的效果如图 11-5 所示。

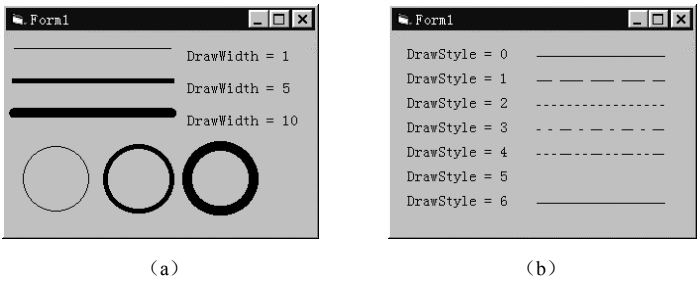


图 11-5 DrawWidth 属性与 DrawStyle 属性不同设置值的效果

当 DrawWidth=1 时，DrawStyle 的设置值全部起作用；当 DrawWidth>1 时，DrawStyle 的设置值为 1~4 时，DrawStyle 属性不起作用，此时绘出的都是实线。

3. FillColor 属性和 FillStyle 属性

利用 FillColor 和 FillStyle 属性，可以对已绘制好的封闭图形（如正方形、矩形、圆形等）和 Shape 控件设置填充图案。当需填充图案时，填充的颜色由 FillColor 属性确定，填充的图案样式由 FillStyle 属性确定。

(1) FillColor 属性

其语法格式为：

[ <对象> .] FillColor [= <值> ]

其中，<值> 为可选的长整型数，为该点指定的 RGB 颜色。如果省略，则默认值为 0。

可用 RGB 函数或 QBColor 函数指定颜色。

(2) FillStyle 属性

其语法格式为：

[ <对象> .] FillStyle [= <值> ]

其中，<值> 的取值范围为 0~7，共有 8 种选择：纯色、横条纹、竖条纹、正网格、斜网格等。不同设置值的效果如图 11-6 所示。

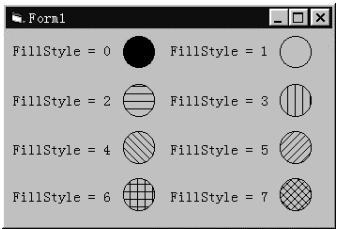


图 11-6 FillStyle 属性不同设置值的效果

11.1.4 使用颜色

1. 颜色属性

窗体和图片框都有两个关于颜色的属性 BackColor 和 ForeColor。BackColor 属性定义了绘画区的背景颜色，若 BackColor 设置为灰色，则清除后整个区域为灰色。ForeColor 属性决定了对象上绘制的文本或图形的颜色，改变 ForeColor 属性，不影响已创建的文本或图形。

在设计时指定颜色属性操作比较简单，只要在属性窗口中单击相应的属性就可以直接利用调色板进行颜色的选择了。而在程序运行中要设置颜色就没有这么直观了，程序员可以使用 VB 预先定义好的颜色常量指定颜色，也可以使用 RGB 函数生成一个颜色。

VB 预先定义好的颜色常量可以用对象浏览器列出，当使用这些内部常数时，无须了解这些常数是如何产生的，也无须声明。例如，无论什么时候想指定红色作为颜色参数或颜色属性的设置值，都可以使用常数 vbRed：

BackColor = vbRed

而颜色函数则提供了更大的选择余地。

2. 颜色函数

VB 提供了两个专门处理颜色的函数 RGB 和 QBColor。

(1) RGB 函数

在这两个颜色函数中，RGB 是最常用的一个，其语法格式为：

RGB (red, green, blue)

其中，red, green, blue 分别表示颜色的红色成分、绿色成分、蓝色成分，取值的范围都是 0~255。

RGB 函数采用红、绿、蓝三原色原理，返回一个长整型数，用来表示一个 RGB 颜色值。

表 11-12 中列出了一些常见的标准颜色，以及这些颜色的红、绿、蓝三原色的成分值。

(2) QBColor 函数

QBColor 函数沿用于早期的 BASIC 版本 QBasic, 返回一个用来表示所对应颜色值的 RGB 颜色码，其语法格式为：

**QBColor (color)**

其中，color 参数是一个介于 0~15 的整型值，见表 11-13。

表 11-12 常见的标准颜色 RGB 值

颜色	红色值	绿色值	蓝色值
黑色	0	0	0
蓝色	0	0	255
绿色	0	255	0
青色	0	255	255
红色	255	0	0
洋红色	255	0	255
黄色	255	255	0
白色	255	255	255

表 11-13 color 参数的设置值

值	颜色	值	颜色
0	黑色	8	灰色
1	蓝色	9	亮蓝色
2	绿色	10	亮绿色
3	青色	11	亮青色
4	红色	12	亮红色
5	洋红色	13	亮洋红色
6	黄色	14	亮黄色
7	白色	15	亮白色

11.1.5 常用绘图方法

使用 VB 提供的绘图方法可以更加灵活地绘制图形。

1. 画点方法 (PSet)

PSet 方法可以在对象的指定位置 (x,y)，按确定的像素颜色画点，其语法格式为：

[〈对象〉.] PSet [Step] (x, y), [〈颜色〉]

【说明】

- ① 〈对象〉为可选的对象表达式，如果省略〈对象〉，则将具有焦点的窗体作为〈对象〉。
- ② Step 为可选的关键字，指定相对于由 CurrentX 和 CurrentY 属性提供的当前图形位置的坐标。
- ③ (x,y) 为必需的一对单精度浮点数，用来设置点的水平和垂直坐标。
- ④ 〈颜色〉为可选的长整型数，为该点指定颜色。如果省略，则使用当前的 ForeColor 属性值。可用 RGB 函数或 QBColor 函数指定颜色。

在 VB 中绘制数学函数曲线多数采用 PSet 方法。

【例 11-2】 用 PSet 方法绘制圆的渐开线，如图 11-7 所示。

【分析】 圆的渐开线可以用以下参数方程表示：

$$\begin{cases} x = a(\cos t + t \sin t) \\ y = a(\sin t - t \cos t) \end{cases}$$

命令按钮的 Click 事件代码如下：

Private Sub Command1\_Click()

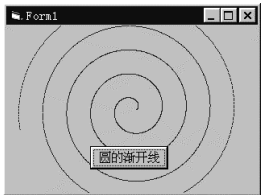


图 11-7 用 PSet 方法绘制圆的渐开线

```

ScaleMode = 6                                ' 设置坐标单位为毫米
x = Me.ScaleWidth / 2                        ' 中心点的横坐标
y = Me.ScaleHeight / 2                      ' 中心点的纵坐标
For t = 0 To 30 Step 0.01
    xt = Cos(t) + t * Sin(t)
    yt = - (Sin(t) - t * Cos(t))            ' 与笛卡儿坐标一致，纵坐标向上为正
    PSet (xt + x, yt + y), vbBlue          ' 相对于中心点画出曲线
Next t
End Sub

```

### 【说明】

① PSet 所画点的大小取决于当前容器（窗体或控件）的 DrawWidth 属性值，像素点的真正颜色取决于 DrawMode 和 DrawStyle 的属性值。

② 执行 PSet 时，CurrentX 和 CurrentY 属性被设置为语句指定的坐标位置。

③ 要清除某个坐标上的像素，只需在该坐标点上画一个颜色为背景色的像素，使用方法如下：

```
PSet(x,y),BackColor
```

## 2. 画直线、矩形方法（Line）

Line 方法可以在对象上的两点之间画直线或矩形，其语法格式为：

〈对象〉.Line [Step] [(x1, y1)] - [Step] (x2, y2) [, 〈颜色〉] [, B [F]]

### 【说明】

① (x1,y1) 为可选项，是直线或矩形的起点坐标。如果省略，起点位于由 CurrentX 和 CurrentY 属性指定的位置。ScaleMode 属性决定了使用的度量单位。

② (x2,y2) 为必需的，是直线或矩形的终点坐标。

③ 〈颜色〉为可选的长整型数，用来设置直线或矩形的颜色。如果省略，则使用 ForeColor 属性值。可用 RGB 函数或 QBColor 函数指定颜色。

④ B 为可选的，如果选择 B，则以 (x1,y1) 为左上角坐标，(x2,y2) 为右下角坐标画出矩形。F 选项用来规定以矩形边框的颜色填充矩形。不能不用 B 而用 F。如果不用 F 仅用 B，则矩形用当前的 FillColor 和 FillStyle 填充。FillStyle 的默认值为 Transparent。

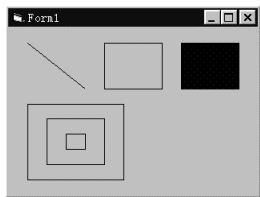


图 11-8 用 Line 方法的不同参数画出的图形

⑤ 画连接的线时，前一条线的终点就是后一条线的起点。线的宽度取决于 DrawWidth 属性值。在背景上画线和矩形的方法取决于 DrawMode 和 DrawStyle 属性值。

⑥ 执行 Line 方法后，CurrentX 和 CurrentY 属性被参数设置为终点。

**【例 11-3】** 下面用 Line 方法的不同参数画图形，如图 11-8 所示。

窗体事件代码如下：

```

Private Sub Form_Paint()
    Cls

```

```

Scale (0, 0)-(13, 11)      ' 设置用户坐标系
Line (1, 1)-(4, 4), 4      ' 画直线
Line (5, 1)-(8, 4), 4, B   ' 画矩形框
Line (9, 1)-(12, 4), 4, BF ' 画矩形块
For i = 1 To 3              ' 画 3 个嵌套的方形框
    Line (i, 4 + i)-(7 - i, 11 - i), , B
Next i

```

**End Sub**

【说明】在使用 PSet 和 Line 方法时，在每个坐标点 (x,y) 之前，可加上 Step 关键字，用来指出将要画出的点和当前坐标点的相对位置，如上例中的第 4, 5, 6 行语句可以用以下语句代替。

```

Line (1, 1)-Step(3, 3), 4      ' 画直线
Line (5, 1)-Step(3, 3), 4, B   ' 画矩形框
Line (9, 1)-Step(3, 3), 4, BF  ' 画矩形块

```

### 3. 画圆方法 (Circle)

Circle 方法可以在对象上画圆、椭圆或弧，其语法格式为：

[〈对象〉.] Circle [Step] (x, y), 〈半径〉, [color, start, end, aspect]

【说明】

① (x,y) 指定圆、椭圆或弧的中心坐标。〈对象〉的 ScaleMode 属性决定了其使用的度量单位。

② 〈半径〉指定圆、椭圆或弧的半径。〈对象〉的 ScaleMode 属性决定了其使用的度量单位。

③ color 可选，如果被省略，则使用 ForeColor 属性值。可用 RGB 函数或 QBColor 函数指定颜色。

④ start 和 end 指定弧或扇形的起点和终点位置（以弧度为单位），其范围为  $-2\pi \sim 2\pi$ 。起点的默认值是 0，终点的默认值是  $2\pi$ 。正数画弧，负数画扇形。

⑤ aspect 为垂直半径与水平半径之比，不能为负数。当  $aspect > 1$  时，椭圆沿垂直方向拉长；当  $aspect < 1$  时，椭圆沿水平方向拉长。aspect 的默认值为 1.0，此时在屏幕上产生一个标准正圆（非椭圆）。

⑥ 可以省略语法中间的某个参数，但不能省略分隔参数的逗号。指定的最后一个参数后面的逗号是可以省略的。

⑦ Circle 执行后，CurrentX 和 CurrentY 属性被参数设置为中心点。

【例 11-4】 利用 Circle 方法在窗体中央画出如图 11-9 所示的图形。

利用 Circle 方法在窗体上先画出一个左上部缺四分之一的圆，然后再用相对坐标在缺口内画第二个圆，最后用小于 1 的纵横比在第二个圆中画一个小椭圆。

窗体的 Click 事件代码如下：

```

Sub Form_Click()
    Const pi = 3.1415926
    Circle (2000, 1250), 1000, vbRed, -pi, -pi / 2
    Circle Step(-500, -500), 500

```

Circle Step(0, 0), 500, , , , 5 / 25

End Sub

【例 11-5】 可以利用 Circle 方法画出部分圆或椭圆，即圆弧和扇形，如图 11-10 所示。

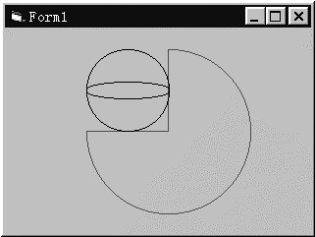


图 11-9 用 Circle 方法在窗体中央画图形

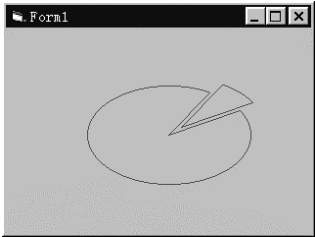


图 11-10 画圆弧

窗体事件代码如下：

Private Sub Form\_Paint()

Const pi = 3.1415926

Circle (2150, 1200), 1000, vbRed, -pi / 6, -pi / 3, 3 / 5

Circle (2000, 1300), 1000, vbRed, -pi / 3, -pi / 6, 3 / 5

End Sub

【说明】画部分圆或椭圆时，如果 start 为负，Circle 画一半径到 start 的圆，并将角度处理为正的；如果 end 为负，Circle 画一半径到 end 的圆，并将角度处理为正的。Circle 方法总是按逆时针（正）方向绘图。

#### 4. 清除图形方法（Cls）

Cls 方法可以清除 Form 或 PictureBox 控件中由图形和打印语句在运行时所生成的图形和文本，清除后的区域以背景色填充。设计时使用 Picture 属性设置的背景位图和放置的控件不受 Cls 方法影响。其语法格式为：

[〈对象〉].Cls

调用 Cls 方法之后，〈对象〉的 CurrentX 和 CurrentY 属性复位为 0。

注意，Cls 方法的使用与 AutoRedraw 属性的设置有很大关系。如果调用 Cls 方法之前，AutoRedraw 属性设置为 False，则 Cls 方法不能清除在 AutoRedraw 属性设置为 True 时产生的图形和文本；如果调用 Cls 方法之前，AutoRedraw 属性设置为 True，则 Cls 方法可以清除所有运行时产生的图形和文本。

【例 11-6】 使用 AutoRedraw 属性控制 Cls 方法从窗体中删除显示信息，如图 11-11 所示。



(a)



(b)

图 11-11 使用 Cls 方法



(1) 用户界面的创建及各对象的属性设置，参见图 11-11。

(2) 编写事件代码。

命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click()  
    Dim Msg, Msg1, Msg2                                ' 声明变量  
    Msg = "由 print 输出的文本"  
    Msg1 = "这是在 AutoRedraw = True 后"  
    Msg2 = "这是在 AutoRedraw = False 后"  
    AutoRedraw = True                                  ' 打开 AutoRedraw 属性  
    CurrentX = ScaleWidth / 2 - TextWidth(Msg) / 2     ' 设置 Print 的 x,y 坐标  
    CurrentY = TextHeight(Msg)  
    Print Msg                                           ' 输出信息 Msg  
    x = ScaleWidth / 2 - TextWidth(Msg1) / 2  
    y = TextHeight(Msg1)  
    Line (x - 50, 4 * y - 50) - Step (TextWidth(Msg1) + 100, y + 100), , B  
    CurrentX = x: CurrentY = 4 * y                     ' 设置 Print 的 x,y 坐标  
    Print Msg1                                          ' 输出信息 Msg1  
    AutoRedraw = False  
    x = ScaleWidth / 2 - TextWidth(Msg2) / 2  
    y = TextHeight(Msg2)  
    Line (x - 50, 7 * y - 50) - Step (TextWidth(Msg2) + 100, y + 100), , B  
    CurrentX = ScaleWidth / 2 - TextWidth(Msg2) / 2     ' 设置 Print 的 x,y 坐标  
    CurrentY = 7 * y  
    Print Msg2                                          ' 输出信息 Msg2  
End Sub
```

命令按钮 Command2 的 Click 事件代码：

```
Private Sub Command2_Click()  
    Msg = "按“确定”按钮，将清除窗体上的部分内容"  
    MsgBox Msg                                          ' 显示信息  
    Cls                                              ' 清除窗体  
End Sub
```

命令按钮 Command3 的 Click 事件代码：

```
Private Sub Command3_Click()  
    AutoRedraw = True : Cls  
End Sub
```

命令按钮 Command4 的 Click 事件代码：

```
Private Sub Command3_Click()  
    Unload Me  
End Sub
```

### 11.1.6 绘图语句与 Paint 事件

如果在程序代码中有图形方法的绘图语句，使用 Paint 事件将很有用。在设计多媒体应用程序时，最有效的方法是将所有的绘图方法（PSet, Line, PaintPicture 等）都放在 Paint 事件中，否则，可能会发生一些不希望发生的事情，例如，图形控件（Label, Line, Shape 等）会重叠、丢失或以错误的顺序排列。

窗体和 PictureBox 图片框控件都有 Paint 事件，通过使用 Paint 事件过程，可以保证必要的图形都得以重现，例如，窗体最小化后，恢复到正常大小时，窗体内所有图形都需要重画。

当 AutoRedraw 属性为 True 时，将自动重画，Paint 事件不起作用。

在 Resize 事件过程中使用 Refresh 方法，可在每次调整窗体大小时强制对所有对象通过 Paint 事件进行重画。

**【例 11-7】** 本例演示 Paint 事件在 Resize 事件中所起的作用。运行时将画出一个与一个窗体各边的中点相交的菱形，当随意调整窗体的大小时，窗体中的菱形也随着自动调整，图 11-12（a）为调整前的显示效果，图 11-12（b）为调整窗体后的显示效果。

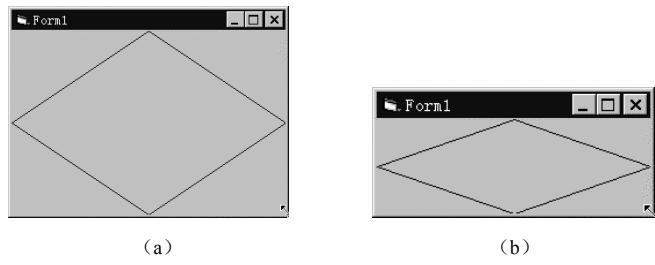


图 11-12 程序运行结果

窗体事件代码如下：

```
Private Sub Form_Paint()  
    Dim HalfX, HalfY                                ' 声明变量  
    HalfX = ScaleLeft + ScaleWidth / 2              ' 设置到窗体宽度的一半  
    HalfY = ScaleTop + ScaleHeight / 2              ' 设置到窗体高度的一半  
    ' 画一个菱形  
    Line (ScaleLeft, HalfY)-(HalfX, ScaleTop)  
    Line - (ScaleWidth + ScaleLeft, HalfY)  
    Line - (HalfX, ScaleHeight + ScaleTop)  
    Line - (ScaleLeft, HalfY)  
End Sub  
Private Sub Form_Resize()  
    Refresh  
End Sub
```

## 11.2 显示图片

在 VB 应用程序中，图片可以显示的 3 个位置有：窗体（Form）上、图片框（Picture）

控件内和图像（Image）控件内。图片可以是下述任何格式的图形图像文件：位图文件（.bmp, .dib, .cur）、图标文件（.ico）、图元文件（.wmf）、增强型图元文件（.emf）、JPEG 或 GIF 文件。图片可来自 Windows 的各种绘图程序，例如，随同各种版本 Windows 一同提供的绘图程序、其他图形应用程序或剪贴画库。可以在设计时或运行时，采用不同途径把图片添加到窗体、图片框或图像控件中。

### 11.2.1 直接加载图片到窗体

使用窗体的 Picture 属性，可以很方便地加载图片到窗体上，如图11-13 所示。要在运行时显示或替换图片，可利用函数 LoadPicture 来设置 Picture 属性：提供图片文件名和可选路径名，由 LoadPicture 函数处理加载和显示图片的细节。

LoadPicture 函数的语法格式为：

```
LoadPicture([ <文件名> ])
```

其中，<文件名>是一个字符串表达式，包括路径和文件的名称。如果省略 <文件名>，LoadPicture 函数将清除图像。



图 11-13 加载图片到窗体

### 11.2.2 使用图像控件

图像控件用来显示图片。实际显示的图片由 Picture 属性决定。Picture 属性中包括了被显示图片的文件名及可选的路径名。

要在运行时显示或替换图片，可利用函数 LoadPicture 来设置 Picture 属性。

图像控件具有 Stretch 属性。Stretch 属性设为 False（默认值）时，图像控件可根据图片调整大小；Stretch 属性设为 True 时，将根据图像控件的大小来调整图片的大小，这可能使图片变形。

图像控件被认为是轻图形控件，它只支持 PictureBox 中属性、方法和事件的一个子集。因此，它需要的系统资源较少而且加载速度也比 PictureBox 控件更快。

**【例 11-8】** 利用图像控件设计“红绿灯”程序，如图 11-14 所示。

设计步骤如下。

（1）建立应用程序用户界面并设置属性。

新建一个工程，进入窗体设计器，在窗体中增加一个框架控件 Frame1。单击 Frame1 使之被激活，在其中加入两个图像控件 Image1 和 Image2。然后在窗体中增加一个图像控件数组 Image3(0)~Image3(2)，如图 11-14 所示。



图 11-14 “红绿灯”程序与界面设计

图像控件的属性设置，参见表 11-14。

表 11-14 属性设置

对 象	属 性	属 性 值
Image1	Picture	trffc10a.ico
	Stretch	True
	Tag	2
Image2	Picture	trffc10c.ico
	Stretch	True
	Tag	3
Image3(0)~Image3(2)	Picture	trffc10a.ico, trffc10b.ico, trffc10c.ico
	Visible	False

(2) 编写事件代码。

编写 Image1 的 Click 事件代码：

```
Private Sub Image1_Click()  
    u = Image1.Tag  
    Select Case u  
        Case 1  
            Image1.Picture = Image3(0).Picture          ' 绿灯  
            Image1.Tag = 2  
            Image2.Picture = Image3(2).Picture  
            Image2.Tag = 1  
        Case 2  
            Image1.Picture = Image3(1).Picture          ' 黄灯  
            Image1.Tag = 3  
        Case 3  
            Image1.Picture = Image3(2).Picture          ' 红灯  
            Image1.Tag = 1  
            Image2.Picture = Image3(0).Picture  
            Image2.Tag = 2  
    End Select  
End Sub
```

编写 Image2 的 Click 事件代码：

```
Private Sub Image2_Click()  
    u = Image2.Tag  
    Select Case u  
        Case 1  
            Image2.Picture = Image3(0).Picture  
            Image2.Tag = 2  
            Image1.Picture = Image3(2).Picture
```

```

Image1.Tag = 1
Case 2
Image2.Picture = Image3(1).Picture
Image2.Tag = 3
Case 3
Image2.Picture = Image3(2).Picture
Image2.Tag = 1
Image1.Picture = Image3(0).Picture
Image1.Tag = 2
End Select

```

**End Sub**

### 【说明】

- ① 3 个图标文件 trffc10a.ico, trffc10b.ico 和 trffc10c.ico 所在文件夹的路径为: \Program Files\Microsoft Visual Studio\Common\graphics\icons\traffic。
- ② 图像控件具有 Click 事件, 因此可以单击图像控件来改变图像的显示。
- ③ 用控件的 Tag 属性记录当前显示的图片。

## 11.2.3 使用图片框控件

图片框 (PictureBox) 控件可以用来显示图片, 作为其他控件的容器, 显示图形方法输出的图形或 Print 方法输出的文本。

### 1. 图片的显示

图片框控件的主要作用是为用户显示图片, 实际显示的图片由 Picture 属性决定。Picture 属性包括了被显示图片的文件名及可选的路径名。

要在运行时显示或替换图片, 可利用函数 LoadPicture 来设置 Picture 属性。

图片框控件具有 AutoSize 属性, 当该属性设置为 True 时, 图片框控件能自动调整自身的大小与显示的图片相匹配。如果要用 AutoSize 属性设置为 True 的图片框控件, 设计窗体时就需要特别小心。图片将不考虑窗体上的其他控件而自动调整大小, 这可能导致意想不到的后果, 如覆盖其他控件等。设计时应通过加载每一幅图片的方法来检查是否会有这种现象发生。

**【例 11-9】** 设计用来浏览图形文件的图片浏览器, 如图 11-15 所示。

设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性。

在窗体中增加文件系统控件: 一个驱动器列表框 Drive1, 一个目录列表框 Dir1, 一个文件列表框 File1, 再增加一个图片框 Picture1, 如图 11-15 所示。

修改 File1 的 Pattern 属性为: “\*.ico; \*.bmp”。

(2) 编写程序代码。

编写目录列表框 Dir1 的 Change 事件代码:

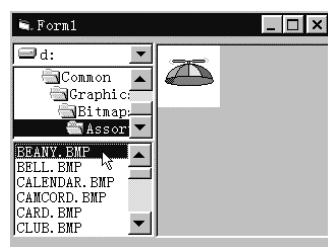


图 11-15 图片浏览器

**Private Sub Dir1\_Change()**

File1.Path = Dir1.Path

**End Sub**

编写驱动器列表框 Drive1 的 Change 事件代码：

**Private Sub Drive1\_Change()**

Dir1.Path = Drive1.Drive

**End Sub**

编写文件列表框 File1 的 Change 事件代码：

**Private Sub File1\_Click()**

ChDrive Drive1.Drive

ChDir Dir1.Path

Picture1.Picture = LoadPicture(File1.FileName)

**End Sub**

【说明】文件系统控件的使用方法参见第 14 章。

2. 输出图形和文本

【例 11-10】 利用图片框输出文本与图形，如图 11-16 所示。

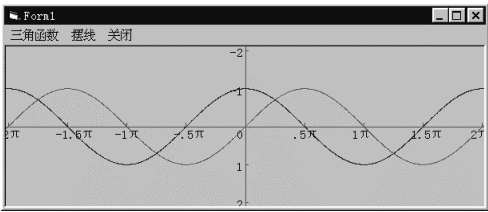


图 11-16 利用图片框输出文本与图形

设计步骤如下。

(1) 首先按照表 11-15 中列出的菜单项设计菜单。

表 11-15 菜单项的设置

标题 (Caption)	名称 (Name)	索引 (Index)	说 明
三角函数	san		主菜单项 1
....Sin(x)	hs1	0	子菜单项 11
....Cos(x)	hs1	1	子菜单项 12
....清除	hs1	2	子菜单项 13
摆线	bai		主菜单项 2
....m = 1.5	m	0	子菜单项 21
....m = 3	m	1	子菜单项 22
....m = 4	m	2	子菜单项 23
....m = 5	m	3	子菜单项 24
....m = 6	m	4	子菜单项 25
....清除	m	5	子菜单项 26
关闭	m		主菜单项 3

然后在窗体上增加一个图片框，如图 11-16 所示。

(2) 编写程序代码。

编写窗体的 Paint 事件代码：

```
Private Sub Form_Paint()  
    Const pi = 3.14159  
    With Picture1  
        .Top = 0  
        .Left = 0  
        .Width = Me.ScaleWidth  
        .Height = Me.ScaleHeight  
        .ScaleMode = 6  
        oldx = .ScaleWidth / 2  
        oldy = .ScaleHeight / 2  
        .Cls  
        ' 画坐标轴  
        Picture1.Line (oldx, 0)-(oldx, .ScaleHeight), RGB(255, 0, 0)  
        Picture1.Line (0, oldy)-(.ScaleWidth, oldy), RGB(255, 0, 0)  
    End With  
    Picture1.CurrentX = oldx - 4: Picture1.CurrentY = oldy + 0.5  
    Picture1.Print 0  
    ' 画 x 轴的刻度  
    For xt = -Int(oldx) To Int(oldx) Step 0.5  
        If xt <> 0 Then  
            st = xt * 10 * pi  
            Picture1.CurrentX = oldx + st - 3: Picture1.CurrentY = oldy + 0.5  
            Picture1.Print xt & "  $\pi$  "  
            Picture1.Line (oldx + st, oldy - 1)-(oldx + st, oldy), RGB(255, 0, 0)  
        End If  
    Next xt  
    ' 画 y 轴的刻度  
    For yt = -5 To 7  
        If yt <> 0 Then  
            st = yt * 10  
            Picture1.CurrentX = oldx - 4: Picture1.CurrentY = oldy + st - 1  
            Picture1.Print yt  
            Picture1.Line (oldx, oldy + st)-(oldx + 1, oldy + st), RGB(255, 0, 0)  
        End If  
    Next yt  
End Sub
```

编写“三角函数”子菜单的 Click 事件代码：

```
Private Sub hs1_Click(Index As Integer)
    oldx = Picture1.ScaleWidth / 2
    oldy = Picture1.ScaleHeight / 2
    Select Case Index
        Case 0
            For t = -oldx To oldx Step 0.01
                xt = 10 * t
                yt = 10 * Sin(t)
                Picture1.PSet (xt + oldx, oldy - yt), RGB(0, 127, 127)
            Next
        Case 1
            For t = -oldx To oldx Step 0.01
                xt = 10 * t
                yt = 10 * Cos(t)
                Picture1.PSet (xt + oldx, oldy - yt), RGB(0, 127, 127)
            Next
        Case 2
            Picture1.Cls
            Form_Paint
            Exit Sub
    End Select
End Sub
```

编写“摆线”子菜单的 Click 事件代码：

```
Private Sub m_Click(Index As Integer)
    n = Index
    Select Case n
        Case 0
            a = 12: b = 8
        Case 1
            a = 12: b = 4
        Case 2
            a = 12: b = 3
        Case 3
            a = 12: b = 2.4
        Case 4
            a = 12: b = 2
        Case 5
            Picture1.Cls
            Form_Paint
            Exit Sub
    End Select
End Sub
```



```

End Select

oldx = Picture1.ScaleWidth / 2
oldy = Picture1.ScaleHeight / 2
For t = 0 To 4 * 3.14159 Step 0.01
    xt = (a + b) * Cos(t) - b * Cos((a + b) * t / b)
    yt = (a + b) * Sin(t) - b * Sin((a + b) * t / b)
    Picture1.PSet (xt + oldx, oldy - yt), vbBlue
Next t

End Sub

```

【说明】内（外）摆线又称“圆内（外）旋轮线”，是 A 圆周沿 B 圆周内（外）部滚动而无滑动时，A 圆周上一固定点 M 所描成的轨迹。其参数方程为：

$$\begin{cases} x = (a \mp b) \cos t \pm b \cos \frac{a \mp b}{b} t \\ y = (a \mp b) \sin t - b \sin \frac{a \mp b}{b} t \end{cases}$$

其中， $a$  为定圆的半径， $b$  为动圆的半径。曲线的形状由  $m = \frac{a}{b}$  的值决定。

本例分别给出当  $m = 1.5, 3, 4, 5, 6$  时的外摆线图形，如图 11-17 所示。

## 习题 11

### 一、选择题

11.1 在下面的属性中，用于自动调整图像框中图形内容的大小的是（ ）。

- A) Picture                      B) CurrentY                      C) CurrentX                      D) Stretch

### 二、填空题

11.2 为了在运行时把 D:\pic 文件夹下的图形文件 a.jpg 装入图片框 Picture1 中，所使用的语句为\_\_\_\_\_。

11.3 下面程序是利用鼠标事件在窗体上画图，如果按下鼠标则可以画图，双击窗体可以清除所画图形。补充完整下面的程序。

首先在窗体层定义如下变量：

```
Dim PaintStart As Boolean
```

编写如下事件过程：

```
Private Sub Form_Load()
```

```
    DrawWith=2
```

```
    ForeColor=vbGreen
```

```
End Sub
```

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

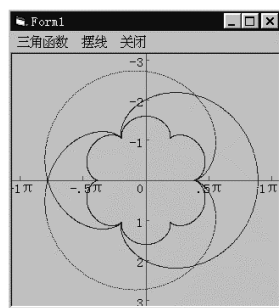


图 11-17 外摆线图形

```
End Sub
Private Sub Form_MouseMove(Button As Integer,Shift As Integer, X As Single,Y As Single)
    If PaintStart Then
        PSet(X,Y)
    End If
End Sub
Private Sub Form_MouseUp(Button As Integer,Shift As Integer, X As Single,Y As Single)

End Sub
Private Sub Form_DblClick()

End Sub
```

三、编程题

- 11.4 编写时钟程序，利用计时器控件来控制指针的转动。
- 11.5 编写程序，实现从蓝色的大圆中分别剪下两个小圆，求大圆剩下部分的面积，如图 11-18 所示。
- 11.6 编写程序，实现输入两点的坐标，可显示两点的连线并计算两点间的距离，如图 11-19 所示。
- 11.7 编写程序，实现在屏幕颜色为 16 位色以上的显示方式下，画一个背景上深下浅的窗体。



图 11-18 习题 11.2 的图

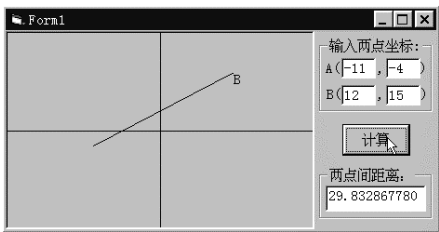


图 11-19 程序界面

- 11.8 利用 Circle 方法在窗体中画一个圆桶，如图 11-20 所示。
- 11.9 利用 Circle 方法在窗体中画一个有缺口的饼，如图 11-21 所示。

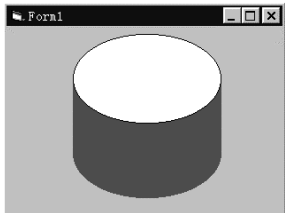


图 11-20 用 Circle 方法在窗体中画一个圆桶

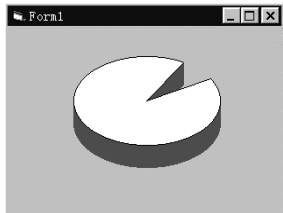


图 11-21 用 Circle 方法在窗体中画一个有缺口的饼

- 11.10 编写程序，输入 3 种商品的销售量，可显示销售比例的饼图，如图 11-22 所示。
- 11.11 在窗体上画五角星，如图 11-23 所示。

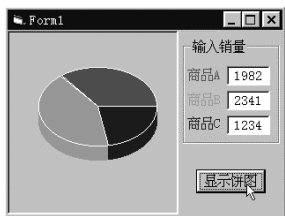


图 11-22 显示销售比例的饼图

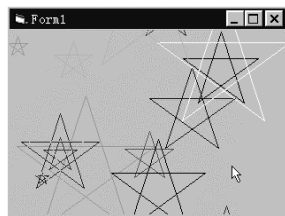


图 11-23 在窗体上画五角星

11.12 编写程序，实现输入长方体的长、宽、高，可求出其表面积。

11.13 编写程序，运行时在图片框中画出坐标系统刻度，在窗体上单击鼠标右键，可在弹出的快捷菜单中选择三角函数名称，可以画出相应的图形。

11.14 编写华氏温度和摄氏温度相互转换的程序，利用文本框交互实现输入和输出。当鼠标指向命令按钮时动态地改变手形状图片的显示，如图 11-24 所示。



图 11-24 动态地改变图片的显示

11.15 编写程序，实现利用滚动条控制图形的大小。

11.16 编写程序，实现利用滚动条控制色彩，还可以返回色彩的 RGB 值。

## 第 12 章 菜单、工具栏与对话框

在 Windows 环境下，几乎所有的应用软件都是通过菜单来提供各种操作的。在 VB 应用程序中，当操作较简单时，一般通过控件来执行；而当要完成较复杂的操作时，使用菜单将更方便。

工具栏同样常常以直观、快捷的特点出现在各种应用程序中，事实上工具栏已经成为 Windows 应用程序的标准功能。它使用户不必在一级一级的菜单中去搜寻需要的命令，给用户带来比菜单更为快捷的操作方式。

在设计 Windows 应用程序时，用户与程序之间的人机交互（如数据的输入和接收、系统信息的反馈等）都是以窗口的形式提供的，这种窗口就是对话框。

### 12.1 菜单

菜单的基本作用有两个：第一是提供人机对话的接口，以便让用户选择应用系统的各种功能；第二是管理应用系统，控制各种功能模块的运行。

一个高质量的菜单程序，不仅要使界面美观，还要方便用户使用，并可避免由于误操作而带来的严重后果。

#### 12.1.1 菜单的两种基本类型

菜单一般分为两种基本类型：下拉式菜单和弹出式菜单。

##### 1. 下拉式菜单

下拉式菜单是一种典型的窗口式菜单，一般通过单击窗口菜单栏中菜单标题的方式打开。例如，在 VB 窗口中单击“文件”、“编辑”等菜单项时所显示的就是下拉菜单。

在下拉式菜单系统中，一般有一个主菜单，即菜单栏（位于窗口标题栏的下方），其中包括一个或多个选择项，称为菜单标题或主菜单项。当单击一个菜单标题时，一个包含若干个菜单项的列表（即菜单）被打开，这些菜单项称为菜单命令或子菜单项。根据功能的不同，菜单命令多以分隔条隔开。有的菜单命令的右端显示有向右的三角符号，当鼠标指针指向该菜单命令时，会出现下级子菜单，VB 中最多可出现 6 级子菜单。有的菜单命令的左边有“√”符号，表示该菜单命令正在起作用。

##### 2. 弹出式菜单

弹出式菜单，也称右键菜单或快捷菜单，是当用户在一个对象上单击右键时显示出来的菜单，可以在窗口的某个位置显示，因此，用户可以利用弹出式菜单更方便快捷地完成操作。在窗体中单击右键时所显示的菜单就是弹出式菜单。

一般地，在不同的对象或区域单击右键，其弹出式菜单的内容是不同的。例如，在 VB

的工具栏中的弹出菜单与窗体设计器中的弹出菜单就完全不同。


### 12.1.2 菜单编辑器

在 VB 中，菜单是一个控件，与其他控件一样，它具有定义其外观与行为的属性。在设计或运行时可以设置 Caption 属性、Enabled 属性和 Visible 属性、Checked 属性，以及其他属性。菜单控件只包含一个事件，即 Click 事件，当用鼠标或键盘选中该菜单控件时，将调用该事件。

与其他控件不同的是，菜单控件不在 VB 的工具箱中，需要在 VB 的菜单编辑器中进行菜单的设计。

#### 1. 菜单编辑器的进入

菜单通过菜单编辑器，即菜单设计窗口建立。可以通过下面 4 种方法进入菜单编辑器：

- 单击“工具”→“菜单编辑器”菜单命令。
- 直接按下快捷键 Ctrl+E。
- 单击工具栏中的“菜单编辑器”按钮 。
- 在要建立菜单的窗体上单击鼠标右键，在快捷菜单中选择“菜单编辑器”命令。

#### 2. 菜单编辑器的组成

进入菜单编辑器后，打开菜单编辑器窗口，如图 12-1 所示。菜单编辑器窗口分为三个部分：菜单属性设置区、编辑区和菜单项显示区。



图 12-1 菜单编辑器

##### (1) 菜单属性设置区

菜单属性设置区用于输入或修改菜单项，并设置菜单项的各个属性。



- 标题：设置菜单项的标题，相当于控件的 Caption 属性，也是显示在菜单中的字符。可以在标题中设置热键，还可以用分隔线将某些菜单项归为一类并与其他项隔开。
- 名称：设置菜单项的名称，相当于控件的 Name 属性。菜单项的命名规则与控件的命名规则相同。
- 索引：设置菜单控件数组的下标，相当于控件数组的 Index 属性。
- 快捷键：设置与菜单项等价的快捷键。在程序运行时，按下快捷键会立刻运行一个菜单项。快捷键的赋值包括功能键与控制键的组合，如 Ctrl+F1 键或 Ctrl+A 键。它们出



现在菜单中相应菜单项的右边。


- 复选：设置为 **True** 时，可以在相应的菜单项旁加上记号“√”，表明该菜单项当前处于活动状态。
- 有效：用来设置菜单项的操作状态。如果该属性被设置为 **False**，则相应的菜单项会变“灰”，不响应用户事件。
- 可见：设置该菜单项是否可见。如果该属性被设置为 **False**，则相应的菜单项将被暂时从菜单中去掉，直到该属性重新被设置为 **True**。


## （2）编辑区


编辑区共有 7 个按钮，用来对输入的菜单项进行简单的编辑。

、：用于产生或取消内缩符号“...”。内缩符号用于确定菜单的层次。单击一次右箭头产生一个内缩符号，单击一次左箭头则删除一个内缩符号。

、：用于调整菜单项的上下位置。当位于菜单控件列表框中的菜单项被选中后，可以通过上、下箭头来移动其位置。

：用于进入下一个菜单项的设计。

：在光标所在处插入一个空白菜单项。

：删除光标所在处的菜单项。

## （3）菜单项显示区

菜单项显示区位于菜单编辑器的下部，输入的菜单项在这里显示出来，并通过内缩符号表明菜单项的层次。

### 【说明】

① 菜单项包括 4 个方面的内容：菜单名、菜单命令、分隔线和子菜单。

② 在输入菜单项时，如果在字母前加上符号“&”，则显示菜单时在该字母下面加一条下划线，可以通过 Alt 键与“带下划线的字母”键组合成快捷键来打开菜单或执行相应的菜单命令。

③ 内缩符号由 4 个点组成，它表明菜单项所在的层次。一个内缩符号（4 个点）表示一层，两个内缩符号（8 个点）表示两层，……，最多可有 6 层。如果一个菜单项前面没有内缩符号，则该菜单为菜单名，即菜单的第一层。

④ 如果在“标题”框内只输入一个“-”符号，则表示产生一个分隔线。

⑤ 只有菜单名没有菜单项的菜单称为“顶层菜单”，在输入这样的菜单项时，通常在后面加上一个感叹号“!”。

⑥ 除分隔线外，所有的菜单项都可以接受 Click 事件。

## 12.1.3 设计下拉式菜单

### 1. 设计下拉菜单的步骤

利用菜单编辑器可以在窗体中建立下拉式菜单，设计步骤如下：

- （1）新建一个窗体，并设计用户界面；
- （2）利用菜单编辑器设计各菜单项；
- （3）利用代码编辑窗口编写每一个菜单项的事件过程；

(4) 运行调试各菜单命令。

2. 下拉菜单设计示例

【例 12-1】 在窗体上建立如图 12-2 所示的下拉式菜单。




图 12-2 建立下拉式菜单

设计步骤如下。

(1) 执行“工具”→“菜单编辑器”菜单命令，打开菜单编辑器。

(2) 在“标题”框中输入“文件(&F)”，在菜单项显示区中将出现同样的标题名称。在“名称”框中输入 file，此时菜单项显示区中没有变化。

(3) 单击编辑区中的“下一个”按钮，菜单项显示区中的条形光标下移，同时菜单属性设置区的“标题”框及“名称”框被清空，光标回到“标题”框。


(4) 在“标题”框中输入“新建”，该信息同时在菜单项显示区中显示出来。在“名称”框中输入“new”。单击编辑区的右箭头 ，菜单显示区中的“新建”右移，同时其左侧出现一个内缩符号“...”，表明“新建”是“文件”的下一级菜单。

(5) 依次输入各菜单项，如果需要指定快捷键，可以单击“快捷键”框右端的下拉箭头，从中选择。例如，为“粘贴”菜单项选中“Ctrl+V”作为其快捷键，如图 12-3 所示。

(6) 设计完成后的窗口如图 12-3 所示，单击“确定”按钮，完成菜单的建立工作。



图 12-3 在菜单编辑器中建立下拉菜单

(7) 单击工具栏中的“运行”按钮 ，运行结果如图 12-2 所示。

【例 12-2】 通过下拉式菜单，改变文本的字体和字型。

设计步骤如下。

(1) 建立用户界面并设置对象属性。

打开菜单编辑器，设置菜单项，参见表 12-1。

表 12-1 设置菜单项

标题 (Caption)	名称 (Name)	说 明
字体	ziti	主菜单项 1
...宋体	songti	子菜单项 11
...楷体	kaiti	子菜单项 12
...黑体	heiti	子菜单项 13
字型	zixing	主菜单项 2
...粗体	cuti	子菜单项 21
...斜体	xieti	子菜单项 22
....下划线	xiahuaxian	子菜单项 23

菜单编辑器的设置如图 12-4 所示。设置了对象属性和菜单项后的用户界面如图 12-5 所示。



图 12-4 菜单编辑器

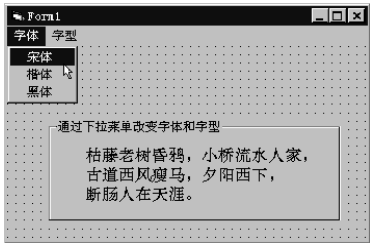


图 12-5 用户界面

(2) 编写菜单项代码。

编写“字体”中 3 个子菜单项的 Click 事件代码：

```
Private Sub songti_Click()  
    Label1.FontName = "宋体"  
End Sub  
Private Sub kaiti_Click()  
    Label1.FontName = "楷体_GB2312"  
End Sub  
Private Sub heiti_Click()  
    Label1.FontName = "黑体"  
End Sub
```

编写“字型”中 3 个子菜单项的 Click 事件代码：

```
Private Sub cuti_Click()  
    cuti.Checked = Not cuti.Checked  
    Label1.FontBold = cuti.Checked  
End Sub
```



```
Private Sub xieti_Click()  
    xieti.Checked = Not xieti.Checked  
    Label1.FontItalic = xieti.Checked  
End Sub  
Private Sub xiahuaxian_Click()  
    xiahuaxian.Checked = Not xiahuaxian.Checked  
    Label1.FontUnderline = xiahuaxian.Checked  
End Sub
```

运行程序，结果如图 12-6 所示。

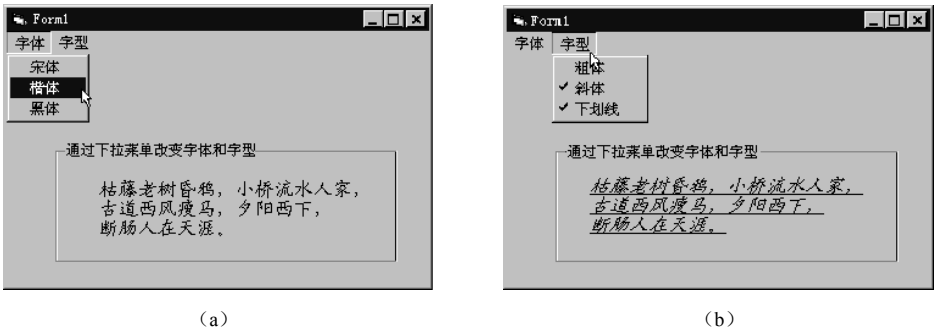


图 12-6 通过下拉菜单改变字体和字型

### 3. 菜单控件数组

由于 VB 将菜单项视为控件，因此就能运用控件数组的概念了。菜单控件数组的作用主要有两个：一是用于动态地增删菜单项；二是简化编程，用一段代码处理多个菜单项。

另外，VB 设计的菜单可以根据程序的运行状态动态地进行调整。当菜单项所指示的操作不适合当前的环境时，可以暂时将其关闭，不让用户选择该菜单项，也可以干脆把它隐藏起来，就像根本没有这个菜单项一样，等到条件成熟时，再重新显示被隐藏的菜单项。

**【例 12-3】** 设计菜单程序。当文本框中没有任何文字时，“字号”菜单中的各项均显示为灰色，表示当前不可用，如图 12-7（a）所示。当用户在文本框中输入了文字后选择某菜单项，可将文字大小设为对应值，并在当前活动项的前面加一个“√”，如图 12-7（b）所示。如果用户选择了“14”项，“10”项将被隐藏，并添加菜单项“16”，其功能与其他菜单项相同，如图 12-7（c）所示。当用户再选择“12”项后，“16”项将被删除，并恢复“10”项的可见性。即文字字号最大为 16，最小为 10，且菜单中只能同时存在 3 个选项。

设计步骤如下。

（1）建立用户界面。

在窗体中添加一个标题为“字号”的菜单，其中包含由“10”、“12”、“14”这 3 个选项组成的菜单控件数组，索引值分别设为 1, 2, 3，如图 12-8 所示。

在窗体中添加一个文本框，并调整大小。



图 12-7 设计文本编辑菜单



图 12-8 设计“字号”菜单

(2) 设置对象属性。

各菜单项的属性设置参见表 12-2。

表 12-2 设置菜单项

标题 (Caption)	名称 (Name)	索引 (Index)	说 明
字号	Main		主菜单项
....10	Size	1	菜单项 1
....12	Size	2	菜单项 2
....14	Size	3	菜单项 3

将文本框 Text1 的 MultiLine 属性设为 True，即允许多行显示；将 ScrollBars 属性设为 2（带垂直滚动条）。

(3) 编写事件代码。

编写菜单标题“字号”的 Click 事件代码：

```
Private Sub Main_Click()  
    If Text1.Text = "" Then  
        Size(1).Enabled = False  
        Size(2).Enabled = False  
        Size(3).Enabled = False  
    Else  
        Size(1).Enabled = True  
    End If  
End Sub
```

```

Size(2).Enabled = True
Size(3).Enabled = True
End If

```

### End Sub

编写各菜单项的 Click 事件代码:

### Private Sub Size\_Click(Index As Integer)

```

Select Case Index
Case 1                                ' 单击“10”项时
    Size(3).Checked = False
    Size(2).Checked = False
    Size(1).Checked = True
    Text1.FontSize = 10
Case 2                                ' 单击“12”项时
    Size(1).Visible = True            ' 被隐藏的“10”项再现
    Size(1).Checked = False
    Size(3).Checked = False
    Size(2).Checked = True
    Text1.FontSize = 12
    If a = 1 Then
        Unload Size(4)                ' 若“16”项已装载，则删除“16”项
        a = 0
    End If
Case 3                                ' 单击“14”项时
    Size(2).Checked = False
    Size(1).Checked = False
    Size(3).Checked = True
    Text1.FontSize = 14
    Size(1).Visible = False
    If a = 0 Then                      ' 若“16”项未装载，则添加新菜单项 Size(4)
        Load Size(4)
        a = 1
        Size(4).Visible = True
        Size(4).Caption = "16"        ' 将新添加项的标题设为“16”
    Else
        Size(4).Checked = False
    End If
Case 4                                ' 单击“16”项时
    Size(2).Checked = False
    Size(3).Checked = False
    Size(4).Checked = True
    Text1.FontSize = 16

```

End Select

End Sub

运行程序，结果如图 12-7 所示。

12.1.4 设计弹出式菜单

设计弹出式菜单的步骤可以分为以下两步。

（1）使用菜单编辑器建立菜单，此步骤与前面介绍的建立下拉菜单的方法一样，只是必须把主菜单的“可见”栏（Visible 属性）设置为 False，其子菜单项的 Visible 属性不要设置为 False。

（2）利用窗体的 PopupMenu 方法显示弹出式菜单。

1. 修改 Visible 属性

对于例 12-2 中的菜单和序，在菜单编辑器中，选择主菜单项“字体”，如图 12-9 所示，单击取消“可见”复选框前面的“√”标记（在默认状态下该项为选中状态）。



图 12-9 在菜单编辑器中取消“可见”复选框的选中状态，修改 Visible 属性

2. PopupMenu 方法

不管是在窗口顶部菜单条上显示的菜单，还是隐藏的菜单，都可以用 PopupMenu 方法把它们作为快捷菜单在程序运行期间显示出来，其语法格式为：

[ <窗体名> .] PopupMenu <菜单名> [, Flags [,x [, y [, Boldcommand ]]]]

【说明】

- ① 若省略 <窗体名> 将打开当前窗体的菜单。
- ② <菜单名> 是指通过菜单编辑器设计的菜单（至少有一个子菜单项）的名称（Name）。
- ③ Flags 参数为一些常数，包含位置和行为两个指定值，见表 12-3、表 12-4。

表 12-3 位置常数

位置常数	说明
0（默认）	菜单左上角位于 X 处
4	菜单上框中央位于 X 处
8	菜单右上角位于 X 处

表 12-4 行为常数

行为常数	说明
0（默认）	菜单命令只接受右键单击
2	菜单命令可接受左、右键单击

两个常数可以相加或以 Or 相连。

④ **Boldcommand** 参数可以指定在显示的弹出式菜单中想以粗体出现的菜单项的名称。

在弹出式菜单中只能有一个菜单项被加粗。

⑤ 为创建一个不显示在菜单栏里的菜单，可在设计时使顶级菜单项目为不可见（保证在菜单编辑器里的“可见”复选框没有被选中）。当 VB 显示一个弹出式菜单时，指定的顶级菜单的 Visible 属性会被忽略。

### 3. 弹出式菜单使用示例

**【例 12-4】** 利用弹出式菜单改变字体，如图 12-10 所示。

设计步骤如下。

（1）在例 12-2 的基础上，建立用户界面并设置对象属性，如图 12-10 所示。

（2）在菜单编辑器中，取消“字体”菜单的“可见”复选框的选中状态，如图 12-11 所示。

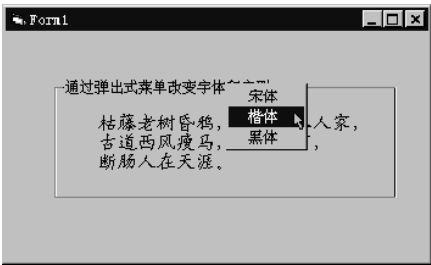


图 12-10 自定义的弹出式菜单



图 12-11 取消“可见”复选框的选中状态

（3）编写事件代码。

编写菜单项的 Click 事件代码：

```
Private Sub songti_Click()  
    Label1.FontName = "宋体"  
End Sub  
Private Sub kaiti_Click()  
    Label1.FontName = "楷体_GB2312"  
End Sub  
Private Sub heiti_Click()  
    Label1.FontName = "黑体"  
End Sub
```

编写框架 Frame1 的 MouseDown（鼠标按下）事件代码：

```
Private Sub Frame1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)  
    If Button = 2 Then  
        PopupMenu ziti, 4 Or 2  
    End If  
End Sub
```

在程序运行时，右键单击框架，即可弹出快捷菜单，如图 12-10 所示。

【例 12-5】 如图 12-12 所示，为文本框增加一个弹出式菜单，该菜单中包含“红色”、“蓝色”和“绿色”3 个选项，单击相应的选项后可以改变文本框中文字的颜色。

设计步骤如下。

(1) 建立用户界面。

添加一个文本框控件 Text1。在菜单编辑器中添加一个标题为“颜色”、名称为 Color 的主菜单。向其中添加“红色”、“蓝色”和“绿色”3 个菜单项。

将顶级菜单的 Visible 属性设为 False（将“可见”复选框前面的“√”去掉，使其不可见），如图 12-13 所示。

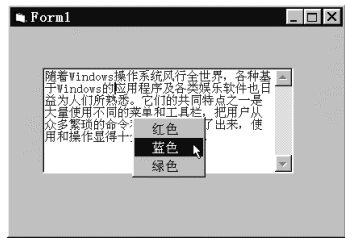


图 12-12 用弹出式菜单改变文本颜色



图 12-13 设计弹出式菜单

(2) 编写事件代码。

编写文本框 Text1 的 MouseDown 事件代码：

```
Private Sub Text1_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
    If Button = 2 Then ' 在文本框中单击右键，弹出快捷菜单
        PopupMenu Color, 4 Or 2 ' Flags 的两个参数值用 Or 运算符连接
    End If
End Sub
```

编写“红色”菜单项 Red 的 Click 事件代码：

```
Private Sub Red_Click()
    Text1.ForeColor = vbRed ' 将文本框中的文字颜色设为红色
End Sub
```

编写“蓝色”菜单项 Blue 的 Click 事件代码：

```
Private Sub Blue_Click()
    Text1.ForeColor = vbBlue ' 将文本框中的文字颜色设为蓝色
End Sub
```

编写“绿色”菜单项 Green 的 Click 事件代码：

```
Private Sub Green_Click()
    Text1.ForeColor = vbGreen ' 将文本框中的文字颜色设为绿色
End Sub
```

运行程序，结果如图 12-12 所示。

#### 4. 文本框中的默认弹出式菜单

其实，在 VB 的文本框中，在不编程的情况下，也可得到一个弹出式菜单。例如，在文本框中，单击鼠标右键，VB 会弹出其默认的弹出式菜单，如图 12-14 所示。

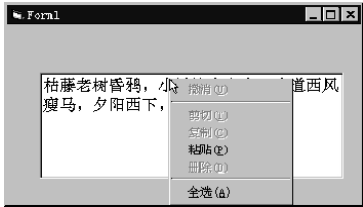


图 12-14 文本框中 VB 默认的弹出式菜单

## 12.2 工具栏

在 VB 中，建立工具栏的方法有两种：手工方式和使用工具栏控件。

### 12.2.1 手工方式设计工具栏

手工方式设计工具栏，就是设计一个图片框，在该图片框中放置一些工具按钮。

#### 1. 手工方式设计工具栏的步骤

手工制作工具栏的一般步骤如下。

- (1) 在窗体上添加一个图片框，将该图片框作为工具按钮的容器（工具栏）。
- (2) 设置图片框的 **Align** 属性以便控制图片框在窗体中的位置。当改变窗体的大小时，图片框（**Align** 属性值非 0）会自动地改变大小以适应窗体的宽度或高度。
- (3) 选定图片框，在图片框中添加需要在工具栏中显示的控件。通常使用的控件有：命令按钮、图形方式的单选钮和复选框、下拉列表框等。
- (4) 设置控件属性。通常在工具按钮上显示不同的图形来表示对应的功能，还可以设置按钮的 **ToolTipText** 属性为工具按钮添加工具提示。
- (5) 编写代码。由于工具按钮通常用于提供对其他（菜单）命令的快捷访问，所以一般在其 **Click** 事件代码中调用对应的菜单命令。

#### 2. 手工方式设计工具栏示例

**【例 12-6】** 如图 12-15 所示，为文本框添加一个简单的工具栏。通过单击工具栏中的按钮，改变文本字体的大小。

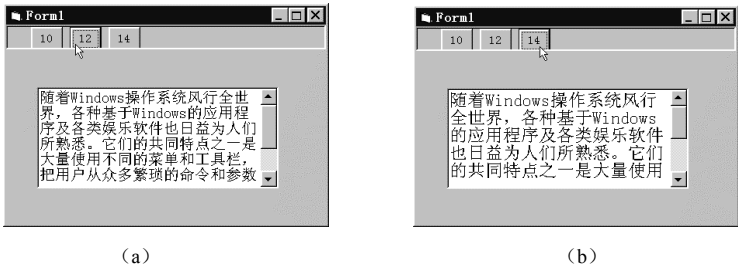


图 12-15 通过工具栏改变字体的大小

设计步骤如下。

(1) 建立用户界面。在窗体中添加一个作为容器使用的图片框，在其中添加由 3 个命令按钮组成的按钮组 Command1(0)~Command1(2)，再增加一个文本框控件 Text1。

(2) 设置对象属性。

将图片框的 Align 属性设为 1 (图片框贴于窗体的顶部)；按钮的 Caption 属性分别设为 10、12 和 14；文本框 Text1 的 MultiLine 属性设为 True (即允许多行显示)，ScrollBars 属性设为 2 (带垂直滚动条)。

(3) 编写事件代码。

编写命令按钮 Command1 的 Click 事件代码：

```
Private Sub Command1_Click(index As Integer)  
    n = index  
    Select Case n                ' 单击命令按钮时调用对应的菜单命令  
        Case 0  
            Text1.FontSize = 10  
        Case 1  
            Text1.FontSize = 12  
        Case 2  
            Text1.FontSize = 14  
    End Select  
End Sub
```

运行程序，结果如图 12-15 所示。

## 12.2.2 使用工具栏控件设计工具栏

使用工具栏 (Toolbar) 控件可以使工具栏的设计更加标准化。

### 1. 添加工具栏控件

工具栏控件是 VB 专业版和企业版所特有的 ActiveX 控件，可以将其添加到工具箱中，以便在工程中使用。

添加工具栏控件的方法为：

(1) 选择“工程”→“部件”菜单命令，打开“部件”对话框；

(2) 在“部件”对话框中，选中“Microsoft Windows Common Controls 6.0”项，单击“确定”按钮。

这时，已在工具箱中增加了一组控件，如图 12-16 所示，其中用来创建工具栏的控件是工具栏控件与图像列表 (ImageList) 控件。

### 2. 工具栏控件的使用方法

工具栏控件的使用方法如下。

(1) 双击工具栏控件按钮，它将被自动添加到窗体中并出现在窗体的顶部，也可单击选中工具栏控件后，在窗体中画出控件。设置图片框的 Align 属性以控制工具栏在窗体中的位置。



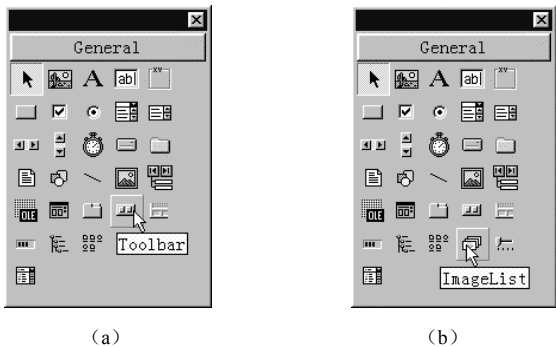


图 12-16 添加到工具箱中的工具栏控件与图图像列表控件

- (2) 右击工具栏控件按钮，在快捷菜单中选择“属性”命令，打开“属性页”对话框。
- (3) 选择“按钮”选项卡，如图 12-17 所示。



图 12-17 “属性页”中的“按钮”选项卡

单击“插入按钮”或“删除按钮”按钮，可以在按钮集合中添加或删除元素。

“标题”与“描述”文本框中是显示在按钮上显示的文字及按钮的说明信息。

在“值”下拉列表中可以设置按钮的状态：0 - tbrUnpressed 为弹起状态，1 - tbrPressed 为按下状态。

在“样式”下拉列表中可以设置按钮的行为特点，并且将影响按钮的功能。

- 0 - tbrDefault：普通（默认的）按钮。
- 1 - tbrCheck：复选框按钮。具有按下、放开两种状态。
- 2 - tbrButtonGroup：单选按钮组。
- 3 - tbrSeparator：分隔符。表示将不同组或不同类的按钮分开，如单选按钮组。
- 4 - tbrPlaceholder：占位符。
- 5 - tbrDropDown：下拉式按钮。可以建立下拉式菜单。

### 3. 利用图像列表控件为工具栏添加图片

工具栏按钮本身没有 Picture 属性，不能像其他控件那样用 Picture 属性直接添加按钮上显示的图片。因此，VB 提供了图像列表（ImageList）控件，以实现工具栏按钮图片的载入。

利用图像列表控件为工具栏添加图片的方法如下。

(1) 向工具栏控件所在的窗体中添加图像列表控件。

(2) 右击窗体中的图像列表控件，在快捷菜单中选择“属性”命令，打开“属性页”对话框，如图 12-18 所示。



图 12-18 “属性页”对话框

(3) 单击“图像”选项卡中的“插入图片”按钮，在弹出的“选定图片”对话框中找到所需要的图片，单击“打开”按钮即可将图片添加到图像列表控件中。重复上述操作直到得到所有需要的图片。

(4) 建立工具栏控件与图像列表控件的关联。打开工具栏控件的“属性页”对话框，在“通用”选项卡的“图像列表”选项中选择图像列表控件名，即可建立两者间的关联。

在程序运行时，下述代码也可建立两者间的关联：

```
Private Sub Form_Load()  
    Toolbar1.ImageList = ImageList1  
End Sub
```

(5) 为工具栏按钮载入图片。一旦工具栏控件与图像列表控件建立了关联，工具栏控件“属性页”对话框的“按钮”选项卡中的“图像”选项即变为有效。只需在其中输入图像库中图像的索引号即可将对应的图片添加到按钮上。

4. 使用工具栏控件设计工具栏示例

【例 12-7】 利用工具栏控件制作工具栏，如图 12-19 所示。单击工具栏中的“粗体”、“斜体”、“下划线”按钮，就能执行相应的操作。工具按钮带有对应的提示信息。

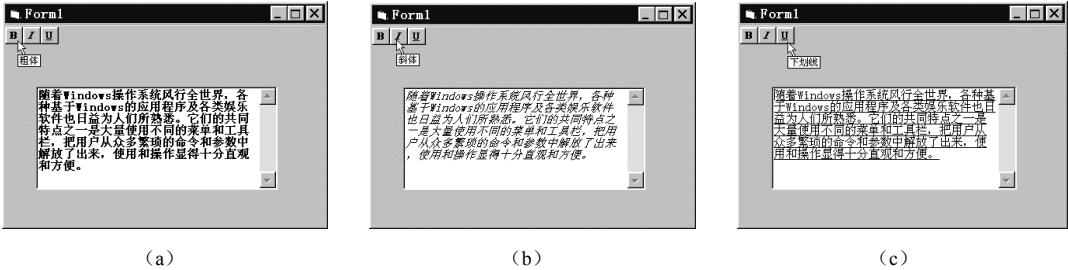


图 12-19 用工具栏控件生成的工具栏

设计步骤如下。

(1) 建立用户界面。

双击工具箱中的工具栏控件按钮向窗体中添加工具栏控件，双击其中的图像列表控件按钮向窗体中添加图像列表控件。

(2) 设置对象属性。

将文本框的 MultiLine 属性设为 True，即允许多行显示；将 ScrollBars 属性设为 2（带垂直滚动条）。

鼠标指向图像列表控件，单击右键，在弹出的快捷菜单中选择“属性”命令，打开“属性页”对话框。选择“图像”选项卡，单击“插入图片”按钮，从“C:\Program Files\Microsoft Visual Studio\Common\Graphics\Bitmaps\TlBr\_W95”文件夹中选择需要的图像，如图 12-20 所示。

在工具栏控件上单击右键，在弹出的快捷菜单中选择“属性”命令，打开工具栏控件的“属性页”对话框。在“通用”选项卡的“图像列表”选项选取 ImageList1，建立与图像列表框的关联。选择“按钮”选项卡，单击其中的“插入按钮”按钮，向工具栏中添加 3 个工具按钮，索引值分别为 1, 2, 3，关键字分别为 B, I, U，对应图像的索引值分别为 1, 2, 3，将工具提示文本分别设为“粗体”、“斜体”、“下划线”，如图 12-21 所示。



图 12-20 向图像列表框中添加图片



图 12-21 建立与图像列表框的关联

(3) 编写事件代码。

编写工具栏 Toolbar1 的 ButtonClick 事件代码：

```
Private Sub Toolbar1_ButtonClick(ByVal Button As MSComctlLib.Button)
    Select Case Button.Index
        Case 1
            Text1.FontBold = True
        Case 2
            Text1.FontItalic = True
        Case 3
            Text1.FontUnderline = True
    End Select
End Sub
```

运行程序，结果如图 12-19 所示。

## 12.3 公共对话框

一些应用程序中常常需要进行打开或保存文件、选择颜色和字体、打印等操作，这就需要应用程序提供相应的对话框以方便使用。这些对话框作为 Windows 的资源，在 VB 中已被做成公共对话框控件。

公共对话框（CommonDialog）控件为用户提供了一组标准的系统对话框，可以使用它进行打开或保存文件、设置打印选项、选择颜色和字体等的操作，另外还可以通过调用 Windows 帮助引擎来显示应用程序的帮助信息。

### 12.3.1 添加公共对话框控件

公共对话框控件属于 VB 专业版和企业版所特有的 ActiveX 控件，位于文件“C:\Windows\System\Comdlg32.ocx”中，名称为“Microsoft Common Dialog Control 6.0”。

右键单击控件工具箱，在弹出菜单中选择“部件”命令，打开“部件”对话框，如图 12-22（a）所示。在“部件”对话框中，选定所需的文件，按“确定”按钮即可将公共对话框控件添加到控件工具箱中，如图 12-22（b）所示。

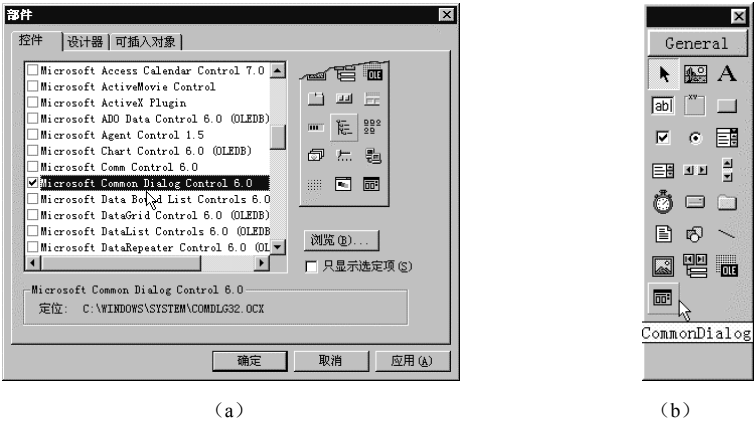


图 12-22 添加公共对话框控件

### 12.3.2 使用公共对话框控件

在应用程序中使用公共对话框控件，需要将它添加到窗体中。由于在程序运行时看不见公共对话框控件，因此可以将它放置在窗体的任何位置。

在程序运行时，公共对话框可以显示一个对话框或执行帮助的引擎，所显示的对话框由控件的“方法”决定，共有 6 种方法来指定相应的对话框，参见表 12-5。

表 12-5 通用对话框控件的方法列表

名 称	功 能	名 称	功 能
ShowOpen	显示文件打开对话框	ShowFont	显示字体对话框
ShowSave	显示文件保存对话框	ShowPrinter	显示打印对话框
ShowColor	显示颜色对话框	ShowHelp	显示 Windows 帮助对话框

每种对话框都有自己特殊的属性，这些属性既可以在属性窗口中设置，也可以在代码中

设置，还可以在“属性页”对话框中设置。“属性页”对话框如图 12-23 所示。

本书介绍表 12-5 中前 5 种对话框的使用方法，帮助对话框的使用可以参见 VB 的联机帮助。

1. 使用“打开”对话框

打开文件是 Windows 应用程序（如 Office）中常用的操作。“打开”对话框可以用来选择文件所在的驱动器、文件夹，以及文件名、文件扩展名，如图 12-24 所示。



图 12-23 “属性页”对话框



图 12-24 “打开”对话框

运行时选定文件并关闭对话框后，可用 FileName 属性得到文件所在的驱动器、文件夹，以及文件名、文件扩展名。

使用“打开”对话框的步骤如下。

- （1）首先在窗体中增加公共对话框控件。
- （2）然后在“属性页”对话框中设置属性，其中各属性描述参见表 12-6。
- （3）最后使用公共对话框控件的 ShowOpen 方法来显示“打开”对话框：  
控件名.ShowOpen

表 12-6 属性页各属性说明

属 性	说 明
对话框标题 (DialogTitle)	用于设置对话框的标题，默认值为“打开”
文件名称 (FileName)	用于设置对话框中“文件名称”的默认值，并返回用户所选中的文件名
初始化路径 (InitDir)	用于设置初始的文件目录，并返回用户所选择的目录。若不设置该属性，默认为当前目录
过滤器 (Filter)	用于设置显示文件的类型，格式为：“描述” “通配符”。若需要设置多项，可以用管道符“ ”隔开，如：All Files (*.*) *.txt Text Files (*.TXT) *.txt
标志 (Flags)	用于设置对话框的一些选项，可以是多个值的组合
缺省扩展名 (DefaultExt)	为该对话框返回或设置默认的文件扩展名，当保存一个没有扩展名的文件时，自动给该文件指定由 DefaultExt 属性指定的扩展名
文件最大长度 (MaxFileSiz)	用于指定文件名的最大字节数。该属性的取值范围为 1~32KB。默认值是 256B
过滤器索引 (FilterIndex)	设置“打开”或“另存为”对话框中默认过滤器的索引。当用 Filter 属性为“打开”或“另存为”对话框指定过滤器时，该属性用来指定默认的过滤器。对于所定义的第一个过滤器，其索引值是 1

2. 使用“另存为”对话框

“另存为”对话框可以用来指定文件所要保存的驱动器、文件夹，以及文件名、文件扩展名，如图 12-25 所示。

使用“另存为”对话框的步骤同“打开”对话框。

最后使用公共对话框控件的 ShowSave 方法来显示“另存为”对话框：  
控件名.ShowSave

3. 使用“颜色”对话框

“颜色”对话框用来在调色盘中选择颜色或创建自定义颜色，如图 12-26 所示。



图 12-25 “另存为”对话框

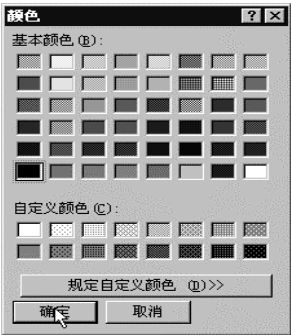


图 12-26 “颜色”对话框

运行时，选定颜色并关闭对话框后，可用 Color 属性得到所选的颜色。使用“颜色”对话框的步骤如下。

- (1) 首先在窗体中增加公共对话框控件。
- (2) 然后在“属性页”对话框中设置属性，其中各属性说明参见表 12-7。

表 12-7 颜色属性说明

属 性	说 明
颜色 (Color)	用于设置初始颜色，并可返回用户所选择的颜色
标志 (Flags)	设置对话框的一些颜色

(3) 最后使用公共对话框控件的 ShowColor 方法来显示“颜色”对话框：  
控件名.ShowColor

4. 使用“字体”对话框

“字体”对话框用来设置并返回所用字体的名字、样式、大小、效果及颜色，如图 12-27 所示。



图 12-27 “字体”对话框

运行时选定设置并关闭对话框后，所做的设置将包含在表 12-8 中。

表 12-8 “字体”对话框确定的属性

名 称	属 性	说 明
颜色	Color	若要使用这个属性，必须先将 Flags 属性设置为 cdICFEffects 或 256
字体样式-粗体	FontBold	
字体样式-斜体	FontItalic	
效果-删除线	FontStrikethru	若要使用这个属性，必须先将 Flags 属性设置为 cdICFEffects 或 256
效果-下划线	FontUnderline	若要使用这个属性，必须先将 Flags 属性设置为 cdICFEffects 或 256
字体	FontName	设置字体的名称
大小	FontSize	设置字体的大小

使用“字体”对话框的步骤如下。

- (1) 首先在窗体中增加公共对话框控件。
- (2) 然后在“属性页”对话框中设置属性，其中各属性说明见表 12-9。

注意，必须将 Flags 属性设为下列常数之一与其他选项之和：

- cdICFScreenFonts 或 1（屏幕字体）
- cdICFPrinterFonts 或 2（打印机字体）
- cdICFBoth 或 3（即 1+2，两种字体皆有）

例如，可设为 259，即 256+3，是 cdICFEffects 常数（256）与 3 之和，在对话框中将出现颜色、效果等选项。

表 12-9 字体属性说明

属 性	说 明
FontName	用于设置字体名称中的初始字体，并可返回用户所选择的字体名称
FontSize	用于设置对话框中的初始字体大小，并可返回用户所选择的字体大小，默认值为 8
Min、Max	用于设置对话框中“大小”列表框中的最小值和最大值
Flags	设置对话框的一些选项
Style	用于设置字体风格，并可返回用户选中的字体风格，它包括 4 个选项：粗体（FontBold）、斜体（FontItalic）、下划线（FontUnderline）、水平删除线（FontStrikethru）

- (3) 最后使用公共对话框控件的 ShowFont 方法来显示“字体”对话框：

控件名.ShowFont

5. 使用“打印”对话框

“打印”对话框可以用来设置打印输出的方法，如打印范围、打印份数、打印质量等。此外，该对话框中还显示了当前安装的打印机的信息，并允许用户重新设置默认打印机。如图 12-28 所示。

使用“打印”对话框的步骤如下。

- (1) 首先在窗体中增加公共对话框控件。
- (2) 然后在“属性页”对话框中设置属性，如图 12-29 所示。



图 12-28 “打印”对话框



图 12-29 “属性页”对话框

其中“属性页”对应的属性说明见表 12-10。

表 12-10 打印属性说明

属 性	说 明
复制 (Copies)	用于设置打印的份数
标志 (Flags)	设置对话框的一些选项。当 Flags 属性为 256 时，将显示“打印设置”对话框
最小 (Min)	用于设置可打印的最小页数
最大 (Max)	用于设置可打印的最大页数
起始页 (FromPage)	用于设置要打印的起始页数
终止页 (ToPage)	用于设置要打印的终止页数
方向 (Orientation)	用于确定以纵向或横向模式打印文档

(3) 最后使用公共对话框控件的 ShowPrinter 方法来显示“打印”对话框：  
控件名. ShowPrinter

12.3.3 公共对话框控件的应用举例

【例 12-8】 这是一个使用公共对话框控件的例子，如图 12-30 所示。  
设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性。

进入窗体设计器，首先增加一个公共对话框 Common Dialog1，一个框架 Frame1 和一个命令按钮数组 Command1 (0)~Command1(3)。然后，选定框架 Frame1，在其中增加一个文本框 Text1。并且参照图 12-30 设置窗体中各控件的属性。

(2) 编写代码。

命令按钮数组 Command()的 Click 事件代码如下：

```
Private Sub Command1_Click(Index As Integer)
    n = Index
    Select Case n
        Case 0
            CommonDialog1.Filter = "所有文件 (*.*)|*. *|文本文件 (*.TXT)|*.txt"
            CommonDialog1.FilterIndex = 1
            CommonDialog1.ShowOpen
            Text1.Text = CommonDialog1.FileName
    End Select
End Sub
```



```

Frame1.Caption = "从打开对话框返回"
Case 1
    CommonDialog1.ShowSave
    Text1.Text = CommonDialog1.FileName
    Frame1.Caption = "从另存为对话框返回"

```

```

Case 2
    CommonDialog1.ShowColor
    Text1.Text = "从颜色对话框返回"
    Text1.ForeColor = CommonDialog1.Color
    Frame1.Caption = "从颜色对话框返回"

```

```

Case 3
    CommonDialog1.Flags = 3 Or 256
    CommonDialog1.ShowFont
    With Text1
        .FontName = CommonDialog1.FontName
        .FontSize = CommonDialog1.FontSize
        .FontStrikethru = CommonDialog1.FontStrikethru
        .FontBold = CommonDialog1.FontBold
        .FontItalic = CommonDialog1.FontItalic
        .FontUnderline = CommonDialog1.FontUnderline
        .ForeColor = CommonDialog1.Color
    End With
    Text1.Text = "从字体对话框返回"
    Frame1.Caption = "从字体对话框返回"

```

End Select

**End Sub**

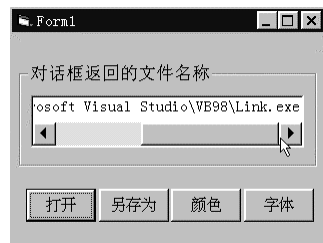


图 12-30 用户界面

## 习题 12

### 一、选择题

12.1 下列说法正确的是（ ）。

- A) 任何时候都可以使用标准工具栏的“菜单编辑器”按钮打开菜单编辑器
- B) 只有当代码窗口为当前活动窗口时，才能打开菜单编辑器
- C) 只有当某个窗体为当前活动窗体时，才能打开菜单编辑器
- D) 任何时候都可以使用“工具”→“菜单编辑器”菜单命令，打开菜单编辑器

12.2 下列叙述错误的是（ ）。

- A) 下拉式菜单和弹出式菜单都用菜单编辑器建立
- B) 在多窗体程序中，每个窗体都可以建立自己的菜单系统
- C) 除分隔线外，所有菜单项都能接受 Click 事件

D) 如果把一个菜单项的 Enabled 属性设置为 False, 则该菜单项不可见

12.3 设菜单中有一个菜单项为“Open”。若要为该菜单命令设置访问键, 即按下 Alt 键及字母 O 时能够执行“Open”命令, 则在菜单编辑器中设置“Open”命令的方式是( )。

- A) 把 Caption 属性设置为&Open
- B) 把 Caption 属性设置为 O&pen
- C) 把 Name 属性设置为&Open
- D) 把 Name 属性设置为 O&pen

12.4 设在菜单编辑器中定义了一个菜单项, 名为 menu1。为了在运行时隐藏该菜单项, 应该使用的语句是( )。

- A) menu1.Enabled=True
- B) menu1.Enabled=False
- C) menu1.Visible=True
- D) menu1.Visible=False

12.5 要使菜单项 MenuOne 在程序运行时失效, 使用的语句是( )。

- A) MenuOne.Visible = True
- B) MenuOne.Visible = False
- C) MenuOne.Enabled = True
- D) MenuOne.Enabled = False

12.6 以下关于菜单的叙述中, 错误的是( )。

- A) 在程序运行过程中可以增加或减少菜单项
- B) 如果把一个菜单项的 Enabled 属性设置为 False, 则可删除该菜单项
- C) 弹出式菜单在菜单编辑器中设计
- D) 利用控件数组可以实现菜单项的增加或减少

12.7 下列有关子菜单的说法中, 错误的是( )。

- A) 除了 Click 事件之外, 菜单项不可以响应其他事件
- B) 每个菜单项都是一个控件, 与其他控件一样也有其属性和事件
- C) 菜单项的索引号必须从 1 开始
- D) 菜单的索引号可以不连续

12.8 在窗体上建立通用对话框需要添加的控件是( )。

- A) Data 控件
- B) From 控件
- C) CommonDialog 控件
- D) VBComboBox 控件

12.9 在窗体上画一个通用对话框, 其名称为 CommonDialog1, 然后画一个命令按钮, 并编写如下事件过程:

```
Private Sub Command1_Click()  
    CommonDialog1.Flags=cdIOFNHideReadOnly  
    CommonDialog1.Filter="All Files(*.*)| *.* | Text Files"& "(*.txt)|*.txt|Batch Files(*.bat)|*.bat"  
    CommonDialog1.FilterIndex=2  
    CommonDialog1.ShowOpen  
    MsgBox CommonDialog1.FileName  
End Sub
```

程序运行后, 单击命令按钮, 将显示一个“打开”对话框, 此时在“文件类型”框中显示的是( )。

- A) All Files(\*.\*)
- B) Text Files(\*.txt)
- C) Batch Files(\*.Bat)
- D) 不确定

12.10 下列程序的功能是调用“字体”对话框来设置文本框字体，单击按钮弹出对话框后，按 Cancel 键退出对话框，则下列说法正确的是（ ）。

```
Private Sub Command1_Click()  
    CommonDialog1.CancelError=true  
    CommonDialog1.flags=cdlCFEffects Or cdlDFBoth  
    CommonDialog1.Action=4  
    CommonDialog1.ShowFont  
    Text1.Font.Name=CommonDialog1.FontName  
    Text1.Font.Size=CommonDialog1.fontSize  
    Text1.Font.Bold=CommonDialog1.FontBold  
    Text1.Font.Italic=CommonDialog1.FontItalic  
    Text1.Font.Underline=CommonDialog1.FontUnderline  
    Text1.Font.Strikethru=CommonDialog1.FontStrikethru  
    Text1.ForeColor=CommonDialog1.Color  
End Sub
```

- A) Text1 的字体不发生变化                      B) Text1 的字体发生变化  
C) Text1 的字体和颜色发生变化                  D) 程序出错！

12.11 以下事件过程可以将“打开”对话框的标题改为“新时代”的是（ ）。

A) **Private Sub Command2\_Click()**  
    CommonDialog1.DialogTitle ="新时代"  
    CommonDialog1.ShowOpen  
**End Sub**

B) **Private Sub Command2\_Click()**  
    CommonDialog1.DialogTitle ="新时代"  
    CommonDialog1.ShowFont  
**End Sub**

C) **Private Sub Command2\_Click()**  
    CommonDialog1.DialogTitle ="新时代"  
    CommonDialog1.Show  
**End Sub**

D) **Private Sub Command2\_Click()**  
    CommonDialog1.DialogTitle ="新时代"  
    CommonDialog1.ShowColor  
**End Sub**

12.12 下列说法错误的是（ ）。

- A) 文件对话框可分为两种，即“打开”(Open)对话框和“另存为”(Save As)对话框  
B) 通用对话框的 Name 属性默认值为 CommonDialogX，此外，每种对话框都有自己的默认标题  
C) “打开”对话框可以让用户指定一个文件，由程序使用；而用“另存为”对话框可以指定一个文件，并以这个文件名保存当前文件

D) DefaultEXT 属性和 DialogTitle 属性都是“打开”对话框的属性，不是“另存为”对话框的属性

12.13 在窗体上画一个名称为 CommonDialog1 的通用对话框，一个名称为 Command1 的命令按钮，要求单击命令按钮时，打开一个“另存为”对话框，该窗口标题为“Save”，默认文件名称为“SaveFile”，在“文件类型”框中显示\*.txt，则能够满足上述要求的程序是( )。

A) Private Sub Command1\_Click()

```
CommonDialog1.FileName="SaveFile"
CommonDialog1.Filter="AllFiles *.* (*.txt) *.txt (*.doc) *.doc"
CommonDialog1.FilterIndex=2
CommonDialog1.DialogTitle="Save"
CommonDialog1.Action=2
```

End Sub

B) Private Sub Command1\_Click()

```
CommonDialog1.FileName="SaveFile"
CommonDialog1.Filter="AllFiles *.* (*.txt) *.txt (*.doc) *.doc"
CommonDialog1.FilterIndex=1
CommonDialog1.DialogTitle="Save"
CommonDialog1.Action=2
```

End Sub

C) Private Sub Command1\_Click()

```
CommonDialog1.FileName="Save"
CommonDialog1.Filter="AllFiles *.* (*.txt) *.txt (*.doc) *.doc"
CommonDialog1.FilterIndex=2
CommonDialog1.DialogTitle="SaveFile"
CommonDialog1.Action=2
```

End Sub

D) Private Sub Command1\_Click()

```
CommonDialog1.FileName=" SaveFile "
CommonDialog1.Filter="AllFiles *.* (*.txt) *.txt (*.doc) *.doc"
CommonDialog1.FilterIndex=1
CommonDialog1.DialogTitle="Save"
CommonDialog1.Action=1
```

End Sub

## 二、填空题

12.14 菜单编辑器可分为 3 个部分，即菜单属性设置区、\_\_\_\_和菜单项显示区。

12.15 在菜单编辑器中建立一个菜单，其主菜单项的名称为 mnuEdit，Visible 属性为 False。程序运行后，如果用鼠标右键单击窗体，则弹出与 mnuEdit 对应的菜单，以下是实现上述功能的程序，请补全程序。

**Private Sub Form\_\_\_\_\_ (Button As Integer,Shift As Integer,X As Single,Y As Single)**

```
If Button=2 Then
    _____ mnuEdit
End If
```

**End Sub**

12.16 在菜单编辑器中建立了一个菜单，名为 pmenu，用下面的语句可以把它作为弹出式菜单弹出，请补全程序。

```
Form1._____ pmenu
```

12.17 在窗体上加上一个文本控件 PCSTextBox，画一个命令按钮，当单击命令按钮的时候将显示“打开”对话框。设置该对话框，使之只用于打开文本文件，然后在文本控件中显示打开的文件名。请补全程序。

```
Private Sub Command1_Click()
    CommonDialog1.Filter = _____
    CommonDialog1.ShowOpen
    PCSTextBox.Text = _____
End Sub
```

**End Sub**

三、编程题

12.18 设计如图 12-31 所示的菜单程序。“程序”菜单中包含有“Word”、“Excel”和“PowerPoint” 3 个选项，“附件”菜单中包含有“画图”和“游戏”两个选项，而“游戏”子菜单中又包含有“纸牌”和“扫雷”两个选项，“退出”菜单中包含有“关闭计算机”和“重新启动”两个选项。当用户选择了“程序”或“附件”中的某一选项时，应能启动相应的程序。当用户选择了“退出”菜单中的某一选项时，将弹出确认对话框，用户确认后将执行相应的操作。

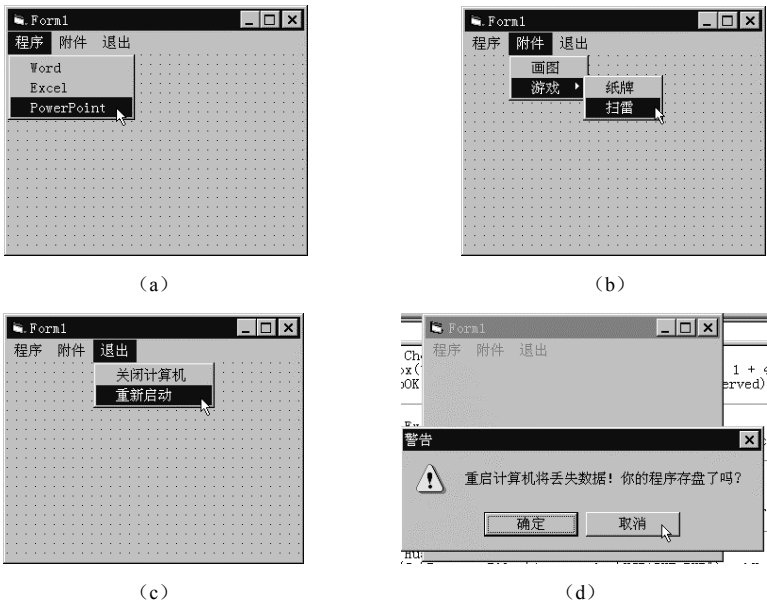


图 12-31 使用菜单的小程序

12.19 编写程序设计个人信息查询工具。要求使用菜单来控制数据的编辑、查找，可完成按姓名查找、按工资范围查找、按工作证号查找的功能。查找时使用模糊查找，将满足条件的记录显示在列表框中。

12.20 设计一个能运行可执行文件（.exe, .com, .bat）的对话框程序，程序启动后界面如图 12-32 所示。单击“浏览”按钮将打开图 12-33 所示的“打开”对话框，在选择文件后单击“打开”按钮，返回对话框程序，此时用户选择的文件名将显示在文本框中。通过选择单选钮可以使程序按“常规”、“最大化”或“最小化”方式运行。单击“取消”按钮将清除文本框中的内容。

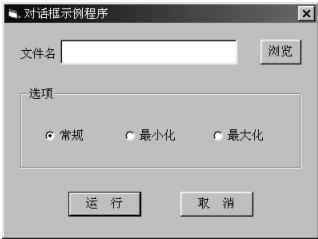


图 12-32 对话框示例程序

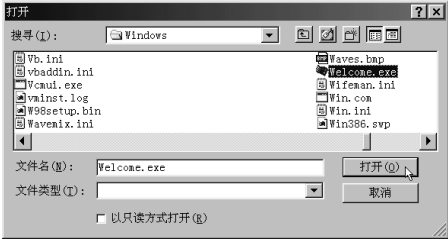


图 12-33 Windows 的“打开”对话框

## 第 13 章 键盘与鼠标事件过程

除了响应鼠标的 Click（单击）或 DblClick（双击）事件以外，VB 应用程序还能响应多种鼠标事件和键盘事件。例如，窗体、图片框与图像控件都能检测鼠标指针的位置，并可判定其左、右键是否已按下，还能响应鼠标键与 Shift 键、Ctrl 键或 Alt 键的各种组合操作。利用键盘事件可以编程响应多种键盘操作，也可以解释、处理 ASCII 字符。

此外，VB 应用程序还可同时支持事件驱动的拖放功能和 OLE 的拖放功能，可用 Drag 方法连同某些属性及事件来启用诸如拖放控件之类的操作。OLE 拖放使应用程序在 Windows 环境下进行数据交换时功能大大增强，无须编写代码就可将其中大多数技术用于应用程序。

### 13.1 键盘事件

键盘事件是指能够响应各种按键操作的 KeyDown、KeyUp 及 KeyPress 事件，通过编写键盘事件的代码，可以响应和处理大多数的按键操作，解释并处理 ASCII 字符。

可以把编写响应键盘事件的应用程序看做是编写键盘处理器。键盘处理器可在控件级（低级）和窗体级这两个层次上工作。有了控件级处理器就可对特定控件编程，例如，可将 Textbox 这个控件中输入的文本都转换成大写字符；而有了窗体级处理器就可使窗体首先响应键盘事件，于是就可将焦点转换成窗体的控件并重复或启动事件。

#### 13.1.1 KeyPress 事件

KeyPress 事件在用户按下和松开一个 ASCII 字符键时发生。该事件被触发时，被按键的 ASCII 码将自动传递给事件过程的 KeyAscii 参数。在程序中，通过访问该参数，即可获知用户按下了哪一个键，并可识别字母的大小写。其语法格式为：

```
Private Sub object_KeyPress(KeyAscii As Integer)
```

其中，参数 KeyAscii 是被按下字符键的标准 ASCII 码，对它进行改变可给对象发送一个不同的字符。将 KeyAscii 改变为 0 时可取消按键，这样一来，对象便接收不到字符了。

##### 【说明】

① 具有焦点的对象才能接收该事件。一个窗体仅在它没有可视的和有效的控件或 KeyPreview 属性被设置为 True 时才能接收该事件。

② KeyPress 事件可以引用任何可打印的键盘字符、来自标准字母表的字符或少数几个特殊字符之一的字符与 Ctrl 键、Enter 键或 Backspace 键的组合。

【例 13-1】 显示按键及其 ASCII 码，界面如图 13-1 所示。

编写复选框的 Click 事件代码：

```
Private Sub Check1_Click()  
    Text1.SetFocus  
End Sub
```

编写文本（输入）框 Text1 的 KeyPress 事件代码：

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
    Text2.Text = KeyAscii
    Select Case KeyAscii
        Case 0 To 7, 9 To 12, 14 To 26, 28 To 31
            Text1.Text = ""
            Text3.Text = "Ctrl+" & Chr(64 + KeyAscii)
        Case 8
            Text3.Text = "Ctrl+" & Chr(64 + KeyAscii) & " 或 退格键"
        Case 13
            Text1.Text = ""
            Text3.Text = "Ctrl+" & Chr(64 + KeyAscii) & " 或 Enter 键"
        Case 27
            Text1.Text = ""
            Text3.Text = "Ctrl+" & Chr(64 + KeyAscii) & " 或 Esc 键"
        Case Else
            Text3.Text = Chr(KeyAscii)
    End Select
    If Check1.Value = 0 Then
        KeyAscii = 0
    End If
End Sub
```



图 13-1 KeyPress 事件接收的按键

【说明】

- ① 函数 Chr(KeyAscii)将 KeyAscii 参数转变为字符。
- ② Text2 中显示按键的 ACSII 码，Text3 中显示按键对应的字符。
- ③ 若取消“回显”项，则 KeyAscii = 0，输入框将不显示刚刚按下的键。

13.1.2 KeyDown 事件和 KeyUp 事件

KeyDown 和 KeyUp 事件是当一个对象具有焦点时按下或松开一个键时发生的。当控制焦点位于某对象上时，按下键盘上的任意一键，则会在该对象上触发产生 KeyDown 事件；



当释放该键时，将触发产生 KeyUp 事件，之后产生 KeyPress 事件。其语法格式为：

```
Private Sub object_KeyDown(KeyCode As Integer, Shift As Integer)
```

```
Private Sub object_KeyUp(KeyCode As Integer, Shift As Integer)
```

其中，参数 KeyCode 用于返回被按键的扫描代码。由于扫描码主要反映物理键位，因此通过该参数不能区分字母大小写，因为同一字母的大小写均是由同一字母键输入的，其扫描码相同。但它可区分主键盘的数字键与小键盘的数字键。

Shift 参数项返回一个整数，该整数反映了 Shift 键、Ctrl 键和 Alt 键的状态。Shift 参数等于 1、2 或 4 分别表示 Shift 键、Ctrl 键或 Alt 键被按下。而 3 个数的部分和表示 3 个按钮部分地被同时按下，例如，若 Ctrl 键和 Alt 键都被按下，则 Shift 的值就是 6。因此，可结合该参数项来判断输入字母的大小写。

【说明】

① 对于这两个事件来说，带焦点的对象都能接收所有按键。一个窗体只有在不具有可见的和有效的控件时才可以获得焦点。

② 虽然 KeyDown 和 KeyUp 事件可应用于大多数键，但是它们最经常地还是应用于：定位键、键盘修饰键和按键的组合，区别小键盘数字键和主键盘数字键，扩展的字符键（如功能键）等。

③ Tab 键不能引用 KeyDown 和 KeyUp 事件；命令按钮的 Default 属性设置为 True 时，Enter 键不能引用 KeyDown 和 KeyUp 事件；命令按钮的 Cancel 属性设置为 True 时，Esc 键不能引用 KeyDown 和 KeyUp 事件。

④ 应当使用 KeyDown 和 KeyUp 事件过程来处理任何不被 KeyPress 识别的按键，诸如：功能键、编辑键、定位键及任何这些键与键盘修饰键（Shift 键、Ctrl 键、Alt 键）的组合等。与 KeyDown 和 KeyUp 事件不同的是，KeyPress 事件不显示键盘的物理状态，而只是传递一个字符。

【例 13-2】 编写一个可以测试功能键与控制键的程序，界面如图 13-2 所示。



图 13-2 测试功能键与控制键

文本（输入）框 Text1 的 KeyDown 事件代码如下：

```
Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)
    Text2.Text = Text2.Text & Str(KeyCode) & ", "
    If KeyCode > 111 And KeyCode < 124 Then
        Label1(2).Caption = "你刚才按了功能键: " & "F" & Str(KeyCode - 111)
        Label1(2).Visible = True
    Else
        Label1(2).Visible = False
    End If
    Check1.Value = IIf((Shift And vbShiftMask) > 0, 1, 0)
    Check2.Value = IIf((Shift And vbCtrlMask) > 0, 1, 0)
    Check3.Value = IIf((Shift And vbAltMask) > 0, 1, 0)
End Sub
```

命令按钮（清除）Command1 的 Click 事件代码如下：

```
Private Sub Command1_Click()
```

```
Text1.Text = ""
```

```
Text2.Text = ""
```

```
Text1.SetFocus
```

```
End Sub
```

#### 【说明】

- ① Text2 中显示接收的 KeyCode 参数，Label1(2)的 Visible 属性设为 False。
- ② 图形方式的复选框 Check1, Check2 和 Check3 分别表示：Shift 键、Ctrl 键和 Alt 键。
- ③ 使用 And 运算符将位屏蔽常数和 Shift 参数一起用来测试条件是否大于 0：

```
ShiftDown = (Shift And vbShiftMask) > 0
```

若大于 0，则说明 Shift 键被按下。

④ KeyPress 事件中将每个字符的大、小写形式作为不同的键代码解释，即作为两种不同的字符。而 KeyDown 和 KeyUp 事件中则用两个参数解释每个字符的大写和小写形式：KeyCode——显示物理的键（将大写字母和小写字母作为同一个键返回）；Shift——指示 Shift+〈Key〉键的状态，并返回大写字母或小写字母其中之一。

### 13.1.3 使用 KeyPreview 属性

KeyPreview 属性返回或设置一个值，以决定是否在控件的键盘事件（KeyDown, KeyUp 或 KeyPress）之前激活窗体的键盘事件。其语法格式为：

```
object.KeyPreview [= Boolean]
```

其中，Boolean 是布尔表达式，指定如何接收事件：当取值为 False（默认值）时，活动控件接收键盘事件，而窗体不接收；当取值为 True 时，窗体先接收键盘事件，然后是活动控件接收事件。

#### 【说明】

- ① 可以用该属性生成窗体的键盘处理程序，例如，应用程序利用功能键时，需要在窗体级处理击键，而不是为每个可以接收击键事件的控件编写程序。
- ② 如果窗体中没有可见的和有效的控件，它将自动接收所有键盘事件。
- ③ 一些控件能够拦截键盘事件，以致窗体不能接收它们，例如，CommandButton 控件有焦点时的 Enter 键，以及焦点在 ListBox 控件上时的方向键。

## 13.2 鼠标事件

在前面的例子中曾多次使用鼠标事件，即 Click（单击）事件和 DblClick（双击）事件，这些事件是通过快速按下并松开鼠标键而产生的。除此之外，VB 还可以通过MouseDown, MouseUp,MouseMove 事件使应用程序对鼠标位置及状态的变化做出响应（其中不包括拖放事件，本章后面的“拖放事件”一节中将介绍拖放事件），大多数控件能够识别这些鼠标事件。

其实，Click 事件是由 MouseDown 事件和 MouseUp 事件组成的，因此 MouseDown 事件和 MouseUp 事件是更基本的鼠标事件。

当鼠标指针位于无控件的窗体上方时，窗体将识别鼠标事件；当鼠标指针在控件上方时，控件将识别鼠标事件。如果鼠标被持续地按下，则第一次按下之后捕获鼠标动作的对象将接收全部鼠标事件直至所有按键被释放为止。

### 13.2.1 MouseDown 事件和 MouseUp 事件

MouseDown 或 MouseUp 事件在按下（MouseDown）或者释放（MouseUp）鼠标按键时发生。其语法格式分别为：

```
Private Sub object_MouseDown(Button As Integer, Shift As Integer, x As Single, y As Single)
```

```
Private Sub object_MouseUp(Button As Integer, Shift As Integer, x As Single, y As Single)
```

其中，参数 Button 返回一个整数，Button 参数的值分别等于 1，2 和 4 时，相应于鼠标左按键、右按键和中间按键的动作。注意，只能有一个按键引起事件。

如同键盘事件一样，参数 Shift 返回一个整数。在 Button 参数指定的按钮被按下或者被释放的情况下，该整数相应于 Shift 键、Ctrl 键或 Alt 键的状态。

参数 x, y 返回一个指定鼠标指针当前位置的数。x 和 y 的值所表示的总是通过该对象的 ScaleHeight, ScaleWidth, ScaleLeft 和 ScaleTop 属性所建立的坐标系统的方式。

【说明】与 Click 和 DblClick 事件不同，MouseDown 和 MouseUp 事件能够区分出鼠标的左按键、右按键和中间按键，也可以为使用 Shift, Ctrl 和 Alt 等键盘换挡键编写用于鼠标-键盘组合操作的代码。为了在给一个鼠标按键按下或释放时指定将引起的一些操作，应当使用 MouseDown 事件或 MouseUp 事件过程。

### 13.2.2 MouseMove 事件

MouseMove 事件在移动鼠标时发生。其语法格式为：

```
Private Sub object_MouseMove(Button As Integer, Shift As Integer, x As Single, y As Single)
```

其中，参数描述同 MouseDown 和 MouseUp 事件。

【说明】

① MouseMove 事件伴随鼠标指针在对象间移动时连续不断地产生。除非有另一个对象捕获了鼠标，否则，当鼠标位置在某个对象的边界范围内时，该对象就能接收 MouseMove 事件。

② 要测试某一条件，首先将各个结果赋给一个临时整型变量，然后再与一个位屏蔽的 Button 或 Shift 参数进行比较。测试时，应当用各个参数进行 And 运算，若结果大于零，则说明该键或按键被按下，其代码如下：

```
LeftDown = (Button And vbLeftButton) > 0
```

```
CtrlDown = (Shift And vbCtrlMask) > 0
```

然后，可对结果的各种组合进行检测，其代码如下：

```
If LeftDown And CtrlDown Then
```

### 13.2.3 自定义鼠标指针

在 VB 中，可以通过属性设置来改变鼠标指针的形状。鼠标指针的改变可以告知用户很

多信息，例如，正在进行长时间的后台任务时，调整某个控件或窗口的大小，某控件不支持拖放操作等。

1. MousePointer 属性的设置

MousePointer 属性是一个整数，取值为 0~15，可用 MousePointer 属性在 16 个预定义指针中任选一个。这些指针表示各种系统事件和过程，表 13-1 中说明了各种指针及其在应用程序中可能的作用。

表 13-1 MousePointer 属性值的说明

指 针 形 状	值	常 数	说 明
	0	vbDefault	（默认值）形状由对象决定
	1	VbArrow	箭头
	2	VbCrosshair	十字线
	3	VbIbeam	I 型
	4	VbIconPointer	图标（矩形内的小矩形）
	5	VbSizePointer	尺寸线（指向东、南、西和北 4 个方向的箭头）
	6	VbSizeNESW	右上-左下尺寸线（指向东北和西南方向的双箭头）
	7	VbSizeNS	垂直尺寸线（指向南和北的双箭头）
	8	VbSizeNWSE	左上-右下尺寸线（指向东南和西北方向的双箭头）
	9	VbSizeWE	水平尺寸线（指向东和西两个方向的双箭头）
	10	VbUpArrow	向上的箭头
	11	VbHourglass	沙漏（表示等待状态）
	12	VbNoDrop	不允许放下
	13	VbArrowHourglass	箭头和沙漏
	14	VbArrowQuestion	箭头和问号
	15	VbSizeAll	四向尺寸线
	99	VbCustom	通过 MouseIcon 属性所指定的自定义图标

每个指针选项均由一个整型设置值表示，默认设置值为 0-Default 并显示成标准的 Windows 箭头指针。但是，此设置由 Windows 操作系统控制，如果用户改变了系统指针的形状，则会改变设置值。为了在应用程序中控制鼠标指针，应将 MousePointer 属性设置为合适的数值。

在设置了控件的 MousePointer 属性后，鼠标经过此控件时，指针就会出现。在设置了窗体的 MousePointer 属性后，鼠标经过窗体的空白区域或经过 MousePointer 属性为 0 - Default 的控件时，选定的指针都会出现。

运行时，可用整型数值或 VB 鼠标指针常数来设置鼠标指针值。例如：

```
Form1.MousePointer = 11      ' 或 vbHourglass
```

2. 图标和光标

用自定义图标或光标可进一步改变应用程序的外观和功能。可以设置鼠标指针来显示自定义图标或光标，它们可以表示鼠标的状态及当前的输入位置。

图标文件的扩展名为.ico，与 VB 中图标文件的扩展名相同。光标文件的扩展名为.cur，

在本质上像图标一样是位图。光标文件中还包含热点信息。热点是跟踪光标位置（ $x$  和  $y$  坐标）的像素。热点通常位于光标的中央。在用 MouseIcon 属性将图标加载到 VB 中后，VB 把它们转换成光标格式并将热点设置成中央像素。两者不同之处是，.cur 文件的热点位置可以改变，而.ico 文件的热点位置不能改变。可在 Windows SDK 提供的 Image Editor 中编辑光标文件。

为使用自定义图标或光标，应设置 MousePointer 和 MouseIcon 属性。其中 MouseIcon 属性设置为自定义图标或光标文件，而 MousePointer 属性则设置成 99 - Custom。

在将 MousePointer 属性设置成 99 - Custom 时，如果未在 MouseIcon 上加载图标，则使用默认的鼠标指针；同样，如果未将 MousePointer 属性设置成 99 - Custom，则忽略 MouseIcon 的设置。

### 13.2.4 使用鼠标事件

**【例 13-3】** 使用鼠标事件设计小画板程序，可以新建或打开已有的.bmp 文件，使用画笔可以随意作图，选择“擦除”项则可以用手形图标擦除画板上的图，还可以保存图形文件，如图 13-3（a）所示。

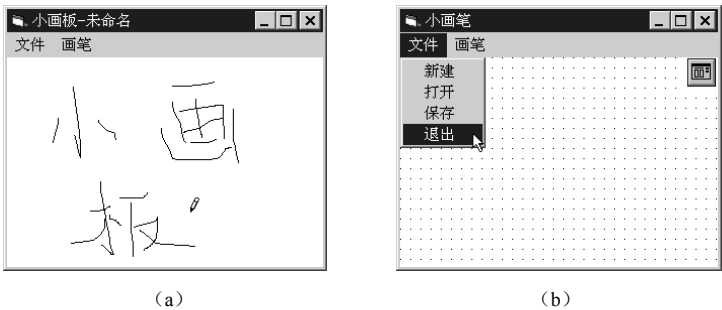


图 13-3 小画板程序

设计步骤如下。

（1）建立应用程序用户界面并设置对象属性。

在窗体中增加公共对话框控件 CommonDialog1，如图 13-3（b）所示。对象属性设置参见表 13-2。

表 13-2 属性设置

对 象	属 性	属 性 值	说 明
Form1	BackColor	（白色）	窗体的背景色
CommonDialog1	Filter	图片文件 *.bmp	过滤器

（2）打开菜单编辑器，按照表 13-3 设计菜单项。

（3）编写程序代码。

编写窗体 Form1 的事件代码。

• Load 事件代码：

```
Private Sub Form_Load()  
    Me.AutoRedraw = True
```

Me.Caption = "小画板-" & "未命名"

End Sub

表 13-3 菜单项的设置

标题 (Caption)	名称 (Name)	索引 (Index)	说 明
文件	File		主菜单项 1
....新建	Files	0	子菜单项 11
....打开	Files	1	子菜单项 12
....保存	Files	2	子菜单项 13
....退出	Files	3	子菜单项 14
画笔	pencil		主菜单项 2

- MouseDown（鼠标按下）事件代码：

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, x As Single, y As Single)
    If Button = 1 Then
        CurrentX = x: CurrentY = y
        End If
End Sub
```

- MouseMove（鼠标移动）事件代码：

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, x As Single, y As Single)
    If Button = 1 Then
        Me.Line (CurrentX, CurrentY)-(x, y)
        CurrentX = x: CurrentY = y
        End If
End Sub
```

编写菜单的事件代码。

- “文件” 菜单的 Click 事件代码：

```
Private Sub files_Click(Index As Integer)
    Select Case Index
        Case 0
            Me.Picture = LoadPicture("")
            Me.Caption = "小画板-" & "未命名"
        Case 1
            CommonDialog1.ShowOpen
            Me.Picture = LoadPicture(CommonDialog1.FileName)
            Me.Caption = "小画板-" & CommonDialog1.FileName
        Case 2
            CommonDialog1.FileName = Mid(Me.Caption, 5)
            CommonDialog1.ShowSave
            SavePicture Me.Image, CommonDialog1.FileName
        Case 3
```

```
End  
End Select
```

**End Sub**

- “画笔” 菜单的 Click 事件代码:

**Private Sub pencil\_Click()**

```
If pencil.Caption = "画笔" Then
```

```
pencil.Caption = "擦除"
```

```
Me.DrawMode = 16
```

```
Me.DrawWidth = 8
```

```
a = "c:\Program Files\Microsoft Visual Studio\Common\Graphics\Cursors\h_nw.cur"
```

```
Else
```

```
pencil.Caption = "画笔"
```

```
Me.DrawMode = 1
```

```
Me.DrawWidth = 1
```

```
a = "c:\Program Files\Microsoft Visual Studio\Common\Graphics\Cursors\Pencil.cur"
```

```
End If
```

```
Me.MouseIcon = LoadPicture(a)
```

**End Sub**

#### 【说明】

- ① 图片文件的存盘命令为:

```
SavePicture Me.Image, CommonDialog1.FileName
```

- ② 设置属性: `Me.AutoRedraw = True` 是为了使图片文件可以接受改动。
- ③ 当鼠标左键被按下时, `CurrentX = x: CurrentY = y` 语句用来保存当时的坐标。

## 13.3 拖放事件

在可视化编程的步骤中, 第一步就是在窗体上摆放一些控件, 这样的操作常常是通过使用鼠标在窗体上拖放控件来实现的。拖放操作其实包括两个操作: “拖” 与 “放”, 即按下鼠标按键并移动控件的操作为 “拖” (动), 释放按键的操作为 “放” (下)。

VB 的拖放功能使应用程序也具有了这种能力, 除了菜单、计时器和通用对话框外, VB 的大多数控件均可在程序运行期间被拖放。

在拖放操作中, 通常把原来位置的对象称为源对象, 拖动后放下的位置的对象称为目标对象。根据具体情况的不同, 可能需要为源对象或目标对象或同时为两者编写代码。

### 13.3.1 与拖放有关的属性、事件与方法

拖放事件是较为复杂的鼠标事件。用下列拖放属性、事件和方法能够指定拖放操作的过程, 而且能指定对于给定控件启动拖动操作的方法。

## 1. 拖放属性

与拖放有关的属性有 DragMode 和 DragIcon。

(1) DragMode 属性用来设置自动或手工拖放模式，取值为 1 或 0，默认值为 0（手工拖放）。该属性可以在属性窗口中设置，也可以在过程代码中设置。

如果 DragMode 属性设置为 1，则使用自动拖放模式。当用户在源对象上按下鼠标左键同时拖动时，该对象的图标会与鼠标指针一起运动。到达目标对象处时放开鼠标，在目标对象上产生一个 DragDrop 事件。需要说明的是，如果未对有关事件编写程序代码，对象本身是不会移动到新的位置上或被加到目标对象中的，只有在目标对象的 DragDrop 事件中进行程序设计，才能实现真正的拖放。

在源对象被拖放到目标对象上的过程中，如果经过其他对象，则将在那些对象上产生 DragOver 事件，同样，对于目标对象也是如此。另外，如果源对象的 DragMode 属性设置为 1，它就不再支持 Click 事件和 MouseDown 事件。

如果 DragMode 属性设置为 0（默认值），则表示启用手动拖放方式。此时，必须在 MouseDown 事件过程中使用 Drag 方法启动“拖”操作。当源对象的 DragMode 设置为 0 时，依然支持 Click 事件和 MouseDown 事件。

(2) DragIcon 属性指定拖动控件时显示的图标。

## 2. 拖放事件

与拖放有关的事件有 DragOver（拖）事件和 DragDrop（放）事件。

(1) DragOver 事件当拖动对象越过一个控件时发生，其过程格式为：

```
Private Sub object_DragOver(Source As Control, x As Single, y As Single, State As Integer)
.....
End Sub
```

其中，参数 Source 的类型为 Control，表示被拖动的对象；参数 x, y 表示拖动时鼠标光标的坐标位置；参数 State 表示鼠标光标与控件的状态，其取值参见表 13-4。

(2) DragDrop 事件当对象被拖到目标位置并松开鼠标按键后发生，其过程格式为：

```
Private Sub Command1_DragDrop(Source As Control, x As Single, y As Single)
.....
End Sub
```

其中，参数 Source 的类型为 Control，表示被拖动的对象；参数 x, y 表示松开鼠标按键放下对象时鼠标光标的坐标位置。

## 3. 拖放方法

与拖放有关的方法是 Drag，其语法格式为：

**〈对象〉.Drag [Action]**

其中，〈对象〉为被拖放的对象；Action 是一个数值，其取值与说明见表 13-5。

### 【说明】

① 只有当对象的 DragMode 属性设置为手工（0）时，才需要使用 Drag 方法控制拖放操作。当然，也可以对 DragMode 属性设置为自动（1 或 vbAutomatic）的对象使用 Drag 方法。



表 13-4 参数 State 的取值

取 值	说 明
0	鼠标光标正进入控件的区域
1	鼠标光标正退出控件的区域
2	鼠标光标正位于控件的区域之内

表 13-5 参数 Action 的取值和说明

常 数	值	说 明
VbCancel	0	取消拖动操作
VbBeginDrag	1	开始拖动对象
VbEndDrag	2	结束拖动对象

② Drag 方法一般是同步的，这意味着其后的语句直到拖动操作完成之后才执行。然而，如果该控件的 DragMode 属性设置为手工，则它可以异步执行。

③ 除 Menu, Timer, Line, Shape 外的所有控件均支持 DragMode 属性、DragIcon 属性和 Drag 方法。

④ 窗体可以识别 DragDrop 和 DragOver 事件，但不支持 DragMode 属性、DragIcon 属性或 Drag 方法。

⑤ 只有当控件没有焦点时才可能被拖动。为防止控件获得焦点，可以将 TabStop 属性设置为 False。

13.3.2 自动拖放

【例 13-4】 幼儿识字练习：拖动图形到方格中，如图 13-4（a）所示。按“对不对？”按钮后，错误的图形将被罩上浅蓝色，如图 13-4（b）所示。按“重新来”按钮，图形回到原处，方格上的标题将重新随机分布。

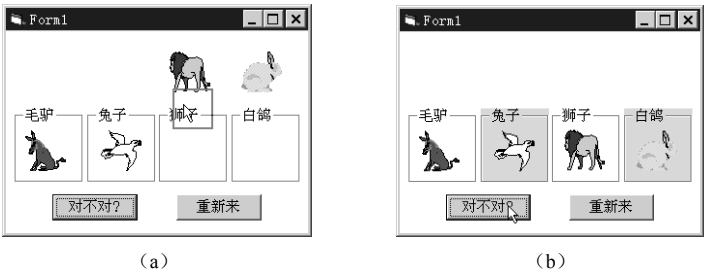


图 13-4 看图识字练习

设计步骤如下。

（1）建立应用程序用户界面并设置对象属性。

在窗体中增加一个图片控件数组 Picture1(0)~Picture1(3)，一个框架控件数组 Frame1(0)~Frame1(3)和两个命令按钮 Command1, Command2。依次选定 Frame1(0)~Frame1(3)，在其中分别增加图像控件数组 Image1(0)~Image1(3)，设置各对象属性参见表 13-6 与图 13-5。

（2）编写程序代码。

首先在通用过程中声明数组：

```
Dim a(4) As String
```

编写窗体 Form1 的 Load 事件代码：

```
Private Sub Form_Load()  
    Randomize Time  
    a(0) = "毛驴": a(1) = "白鸽": a(2) = "狮子": a(3) = "兔子"  
End Sub
```

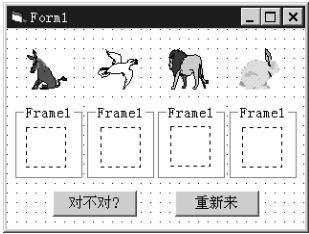


图 13-5 建立界面与设置属性

表 13-6 属性设置

对 象	属 性	属 性 值	说 明
Form1	BackColor	(白色)	窗体的背景色
Frame1()	BackColor	(白色)	框架的背景色
Picture1()	Dragmode	1 - Automatic	拖动模式
	Picture	依次改为: donkey.wmf, dove.wmf, lion.wmf, rabbit.wmf	
	BackColor	(白色)	
	BorderStyle	0 - None	边框
Image1()	Stretch	True	
Command1	Caption	重新来	
Command2	Caption	对不对?	

编写窗体 Form1 的 Activate 事件代码:

```
Private Sub Form_Activate()  
    For i = 0 To 3  
        j = Int(Rnd() * 4)  
        Do While a(j) = ""  
            j = Int(Rnd() * 4)  
        Loop  
        Frame1(i).Caption = a(j): a(j) = ""  
    Next i  
    a(0) = "毛驴": a(1) = "白鸽": a(2) = "狮子": a(3) = "兔子"  
End Sub
```

编写“重新来”按钮 Command1 的 Click 事件代码:

```
Private Sub Command1_Click()  
    Form_Activate  
    For i = 0 To 3  
        Image1(i).Picture = LoadPicture()  
        Picture1(i).Visible = True  
        Frame1(i).BackColor = RGB(255, 255, 255)  
    Next  
End Sub
```

编写“对不对?”按钮 Command2 的 Click 事件代码:

```
Private Sub Command2_Click()  
    For i = 0 To 3  
        If Image1(i).Tag <> Frame1(i) Then  
            Frame1(i).BackColor = RGB(128, 255, 255)  
        End If  
    Next  
End Sub
```

编写图像数组 Image1()的 DragDrop 事件代码:

```
Private Sub Image1_DragDrop(Index As Integer, Source As Control, X As Single, Y As Single)
    Image1(Index).Picture = Source.Picture
    Source.Visible = False
    Image1(Index).Tag = a(Source.Index)
End Sub
```

【说明】

- ① 在窗体的 Load 事件代码中，将图形的名称随机赋值给框架的标题。而将图形拖放到方格中时，又将名称赋予图像的 Tag 属性，以便以后对照。
- ② 图形被拖放到方格中，其实是将图形文件赋予图像控件 Image1，而将原图形隐藏起来 (Source.Visible = False)。

13.3.3 手工拖放

在上面的例子中，为了使控件在拖放的操作中能较为精确地定位，需要使用手动模式进行拖放。

【例 13-5】 修改例 13-4 为手动拖放模式，可以将图形停放在窗体上的任意位置，如图 13-6 所示。

在例 13-4 的基础上进行设计，具体步骤如下。

- (1)修改图片数组 Picture1()的 Dragmode 属性为:  
0 - Manual。

- (2) 增加代码。

首先在通用模块中增加变量声明：

```
Dim oldx(3), oldy(3) As Single
```

修改窗体 Form1 的 Load 事件代码：

```
Private Sub Form_Load()
    Randomize Time
    a(0) = "毛驴": a(1) = "白鸽": a(2) = "狮子": a(3) = "兔子"
    For i = 0 To 3
        ' 增加的循环
        oldx(i) = Picture1(i).Left
        oldy(i) = Picture1(i).Top
    Next
End Sub
```

修改窗体 Form1 的 Activate 事件代码：

```
Private Sub Form_Activate()
    For i = 0 To 3
        j = Int(Rnd() * 4)
        Do While a(j) = ""
            j = Int(Rnd() * 4)
        Loop
    Next
End Sub
```



图 13-6 将图形停放在窗体上的任意位置

```

Frame1(i).Caption = a(j): a(j) = ""
Picture1(i).Move oldx(i), oldy(i)           ' 增加的语句
Next i
a(0) = "毛驴": a(1) = "白鸽": a(2) = "狮子": a(3) = "兔子"

```

**End Sub**

编写窗体 Form1 的 DragOver 事件代码:

```

Private Sub Form_DragOver(Source As Control, X As Single, Y As Single, State As Integer)
    Source.Move (X - Source.Width / 2), (Y - Source.Height / 2)

```

**End Sub**

编写图片数组 Picture1() 的 MouseDown 事件代码:

```

Private Sub Picture1_MouseDown(Index As Integer, Button As Integer, Shift As Integer, _
                                X As Single, Y As Single)
    Picture1(Index).Drag 1

```

**End Sub**

编写图片数组 Picture1() 的 MouseUp 事件代码:

```

Private Sub Picture1_MouseUp(Index As Integer, Button As Integer, Shift As Integer, _
                                X As Single, Y As Single)
    Picture1(Index).Drag 0

```

**End Sub**

### 【说明】

- ① oldx(i), oldy(i) 用来记录图片的初始位置。
- ② 当 MouseDown 事件发生时, 调用控件的 Drag 方法。
- ③ 当控件在窗体上拖动时 (DragOver 事件), 需要随时定位控件的位置。

## 习题 13

### 一、选择题

13.1 在窗体上画一个名称为 TxtA 的文本框, 然后编写如下的事件过程:

```

Private Sub TxtA_KeyPress(KeyAscii As Integer)
    .....
End Sub

```

若焦点位于文本框中, 则能够触发 KeyPress 事件的操作是 ( )。

- |            |              |
|------------|--------------|
| A) 单击鼠标    | B) 双击文本框     |
| C) 鼠标滑过文本框 | D) 按下键盘上的某个键 |

13.2 把窗体的 KeyPreview 属性设置为 True, 然后编写如下事件过程

```

Private Sub Form_KeyPress(KeyAscii As Integer)
    Dim ch As String
    ch=Chr(KeyAscii)
    KeyAscii=Asc(UCase(ch))

```

```
Print Chr(KeyAscii+2)
```

**End Sub**

程序运行后，按键盘上的 A 键，则在窗体上显示的内容是（ ）。

- A) A                      B) B                      C) C                      D) D

13.3 以下叙述中错误的是（ ）。

- A) 在 KeyUp 和 KeyDown 事件过程中，从键盘上输入 A 或 a 被视做相同的字母（即具有相同的 KeyCode）  
B) 在 KeyUp 和 KeyDown 事件过程中，将键盘上的 1 键和右侧小键盘上的 1 键视做不同的数字（具有不同的 KeyCode）  
C) KeyPress 事件中不能识别键盘上某个键的按下与释放  
D) KeyPress 事件中可以识别键盘上某个键的按下与释放

13.4 假定有如下事件过程：

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

```
    If Button=2 Then
```

```
        PopupMenu popForm
```

```
    End If
```

```
End Sub
```

则以下描述错误的是（ ）。

- A) 该过程的功能是弹出一个菜单  
B) popForm 是在菜单编辑器中定义的弹出式菜单的名称  
C) 参数 X、Y 指明鼠标的当前位置  
D) Button=2 表示按下的是鼠标左键

13.5 在窗体上画 1 个文本框，其名称为 Text1，然后编写如下过程：

```
Private Sub Text1_KeyDown(KeyCode As Integer, Shift As Integer)
```

```
    Print Chr(KeyCode)
```

```
End Sub
```

```
Private Sub Text1_KeyUp(KeyCode As Integer, Shift As Integer)
```

```
    Print Chr(KeyCode+2)
```

```
End Sub
```

程序运行后，把焦点移到文本框中，此时如果按下 A 键，则输出结果为（ ）。

- A) A                      B) A                      C) A                      D) A  
A                      B                      C                      D

13.6 在窗体上画一个文本框，然后编写如下事件过程：

```
Private Sub Text1_KeyPress(KeyAscii As Integer)
```

```
    Dim char As String
```

```
    char=Chr(KeyAscii)
```

```
    KeyAscii=Asc(UCase(char))
```

```
    Text1.Text=String(6, KeyAscii)
```

```
End Sub
```

程序运行后，如果从键盘上输入字母 a，则文本框中显示的内容为

- A) a                      B) A                      C) aaaaaaa                      D) AAAAAAA

13.7 在窗体上画一个名称为 Command1 的命令按钮，然后编写如下程序：

```
Dim SW As Boolean

Function func(X As Integer)As Integer
    If X<20 Then
        Y=X
    Else
        Y=20+X
    End If
    func=Y
End Function

Private Sub Form_MouseDown(Button As Integer, Shift As Integer,X As Single,Y As Single)
    SW=False
End Sub

Private Sub Form_MouseUp(Button As Integer, Shift As Integer,X As Single,Y As Single)
    SW=True
End Sub

Private Sub Command1_Click()
    Dim intNum As Integer
    intNum=InputBox(" ")
    If SW Then
        Print func(intNum)
    End If
End Sub
```

程序运行后，单击命令按钮，将显示一个输入对话框，如果在对话框中输入 25，则程序的执行结果为（ ）。

- A) 0                      B) 25                      C) 45                      D) 无任何输出

## 二、填空题

13.8 在 KeyPress 事件过程中，KeyAscii 是所按键的\_\_\_\_值。

13.9 有如下事件过程，当同时按下 Shift 键和 F5 键时，最后输出的信息是\_\_\_\_\_。

```
Const ShiftKey =-1
Const CtrlKey =2
Const Key_F5 =&H74
Const Key_F6 =&H75

Private Sub Text1_KeyDown(KeyCode As Integer,Shift As Integer)
    If KeyCode = Key_F5 And Shift = ShiftKey Then
        Print "Press Shift+F5"
    End If
End Sub
```

```
ElseIf KeyCode = Key_F6 And Shift = CtrlKey Then
    Print "Press Ctrl+F6"
End If
```

End Sub

- A) 无任何信息            B) Press Shift+F5            C) Press Ctrl+F6            D) 程序出错

三、编程题

13.10 编写一个程序，当同时按下 Shift 键和 F6 键时，在窗体上显示“再见！”，并终止程序的运行。

13.11 编写一个程序，当按下某个键时，程序以十六进制和八进制形式输出该键的 KeyCode。

13.12 在窗体上画一个文本框、一个图片框和一个命令按钮。编写程序，使得当鼠标光标位于不同的控件或窗体上时，鼠标光标具有不同的形状，此时如果同时按下鼠标右键，则显示相应的信息。例如，当鼠标光标移动到图片框上时，如果按下鼠标右键，则用一个信息框显示“这是图片框”。

要求：在文本框和窗体上的鼠标光标使用系统提供的光标形状，而图片框和命令按钮上的鼠标光标使用自己定义的形状。

13.13 编写一个类似于“回收站”的程序，并用适当的图形作为“回收站”图标。程序运行后，把窗体上其他的对象拖到“回收站”图标上，松开鼠标键后，显示一个信息框，询问是否要把该对象放入回收站，如图 13-7 所示。此时单击“是”按钮即放入回收站，对象从窗体上消失；单击“否”按钮，则对象仍回到原来位置。

13.14 在窗体上画若干个控件，然后画两个列表框，其中一个列表框用来列出当前窗体上控件的名称，另一个列表框列出 15 种鼠标光标的形状。程序运行后，从第一个列表框中选择控件或窗体，从第二个列表框中选择鼠标光标形状，为选择的控件或窗体设置所需要的鼠标光标形状，如图 13-8 所示。

要求：两个列表框隐藏，只在需要时显示出来。



图 13-7 回收站

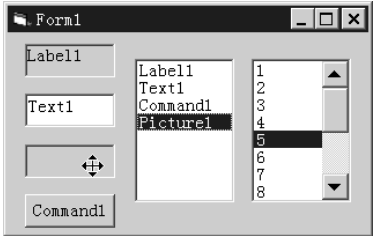


图 13-8 设置鼠标光标形状

## 第14章 数据文件

大多数的应用程序都需要读/写磁盘文件。本章介绍如何建立文件、读取文件、写文件，以及对文件夹中的文件进行删除、复制、更名等各种操作。

### 14.1 文件的分类与结构

文件是指记录在外部介质上的信息的集合，文件的结构是指如何合理地组织数据从而形成文件，当然这与文件的类型有关。

#### 14.1.1 文件的分类

根据不同的分类标准，VB 文件可以分为不同的类型。

##### 1. 按文件性质分类

根据文件的性质，可分为程序文件和数据文件两大类。

- 程序文件：这种文件中存放的是可供计算机执行的程序，包括源程序文件和可执行程序文件。例如，扩展名为.com 和.exe 的可执行文件及扩展名为.bas 和.frm 的源程序文件。
- 数据文件：用来存放运行程序所需的数据，或存储程序的运行结果。例如，学生成绩、职工工资、人事档案等。

本章主要讨论数据文件。

##### 2. 按存取方式和结构分类

根据文件中数据存取方式的不同，可以将数据文件分为顺序文件和随机文件两大类。

- 顺序文件：数据（通常以记录的形式存放）的写入是一个接一个依次进行的。数据在文件中的存放次序及读出次序与写入数据时的顺序一致，也是从头到尾按顺序进行的。这样的文件结构较为简单，但维护困难，为了修改文件中的某条记录，必须把整个文件读入内存，修改完后再重新写入磁盘。不能灵活地存取和增减数据，适用于有一定规律且不经常修改的数据。其优点是占空间少，容易使用。
- 随机文件：数据通常也以记录的形式存放，但与顺序文件不同的是，其每条记录的长度相等，且拥有一个唯一的记录号。因此，对于随机文件，可以按记录号进行数据的存取操作，不但可以随机地访问任意指定的记录，而且对记录的读或写操作也是可以随意选择的。随机文件对数据的存取操作较顺序文件更方便、灵活。

##### 3. 按编码方式分类

根据文件中存储信息所使用的编码方式，可以将文件分为 ASCII 文件和二进制文件。

- ASCII 文件：又称为文本文件，它以 ASCII 方式存储，数值型数据中的每位数字分别



使用代表它们的 ASCII 码存储，汉字的存储则使用双字节的汉字字符集编码。ASCII 文件可以用 DOS 的 TYPE 命令显示，可以直接打印输出，可以用文本编辑软件处理。例如，DOS 编辑器 edit、Windows 中的“记事本”等。高级语言的源程序通常也是 ASCII 文件。

- 二进制文件：以二进制方式保存信息，该类文件不具有可读性，不能使用 TYPE 命令输出或显示，也不能用文本编辑器建立或修改，占空间较小。二进制文件通常用于编译后的程序文件。

### 14.1.2 文件的结构

为了有效地存取数据，数据必须以某种特定的方式存放，这种特定的方式就是文件的结构。VB 的数据文件由记录组成，记录由字段组成，字段由字符组成。

#### (1) 字符 (Character)

字符是构成文件的最基本单位，可以是数字、字母、特殊符号或单一的字节。一个字符通常用一个字节存放，一个汉字或全角字符则用两个字节存放。注意，VB 6.0 支持双字节字符，当计算字符串长度时，一个汉字作为一个字符计算。

#### (2) 字段 (Field)

字段又称为域，由若干个字符组成，用来表示一项数据。例如，“姓名”字段中的“张大强”由 3 个汉字组成。

#### (3) 记录 (Record)

记录由一组相关的字段组成。例如，在通信录中，每个人的姓名、单位、住址、电话号码等信息组成一个记录。在 VB 中，通常以记录为单位处理数据。

#### (4) 文件 (File)

文件由记录组成。一个文件含有一条以上的记录。例如，在通信录文件中有 40 个人的信息，每个人的信息是一条记录，40 条记录构成一个文件。

## 14.2 文件操作语句和函数

在 VB 中，对于数据文件的处理，传统的方法是通过使用 Open 语句及一些相关的语句和函数来实现。这些语句和函数，适用于顺序文件、随机文件和二进制文件的访问。

### 14.2.1 数据文件的操作

在微型计算机中，数据文件一般为磁盘文件。从磁盘文件向计算机的内存传送数据，对于计算机来说，属于“输入”操作，称为“读文件”。从计算机的内存向磁盘文件传送数据，则是计算机的“输出”操作，称为“写文件”。

为了有效地管理文件的输入/输出操作，每一个打开的数据文件中都有一个指针，指向下一次将要读/写的数据位置，称为“文件指针”或“记录指针”。当默认读/写位置时，数据的读出或写入总是指向文件指针的当前位置。

数据文件的操作，一般按以下 3 个步骤进行。

- (1) 打开（或建立）文件。一个数据文件，首先必须打开才能使用。如果文件不存在，

在执行某些打开命令时，将建立一个新文件。

（2）读/写文件。执行文件的“写”操作，就是把内存中的数据传输到外部设备（一般为磁盘）中并予以存储的过程；执行文件的“读”操作，则是把文件中的数据传输到计算机内存的过程。读/写文件是数据文件处理的核心部分。

（3）关闭文件。对于一个不再使用的文件，应执行关闭命令，以便释放相关的文件缓冲区。

## 14.2.2 文件的打开与关闭语句

### 1. Open 语句

在对文件执行任何读/写操作之前，必须打开文件。Open 语句用来打开或建立一个文件，分配一个缓冲区供文件进行输入/输出操作，并决定缓冲区的访问方式。其语法格式为：

```
Open <文件名> For <读写方式> [Access <存取类型>] [ <锁定类型>] As [#] <文件号>
[Len = <记录长度>]
```

【说明】

- ① <文件名> 为欲打开或建立的文件名，其中还可包括驱动器名和路径描述。
- ② <读写方式> 用来指定文件的读/写方式，其取值见表 14-1。

表 14-1 读/写方式

参 数	方 式	说 明
Output	顺序输出方式	打开或建立一个顺序文件，并允许向文件输出数据
Input	顺序输入方式	打开一个顺序文件，指定从文件中读入数据
Append	顺序输出（追加）方式	与 Output 不同的是，Append 方式在打开一个顺序文件时，将指针定位在文件的末尾，当输出数据时，新记录将被添加到原有记录的后面
Random	随机文件方式	如果未指定方式，Open 语句将以 Random 方式打开文件
Binary	二进制文件方式	

- ③ <存取类型> 用来指定文件的存取类型，其取值见表 14-2。

表 14-2 存取类型

参 数	存 取 类 型	说 明
Read	打开只读文件	
Write	打开只写文件	
Read Write	打开读/写文件	只能用于随机文件、二进制文件和以 Append 方式打开的文件。在 Random 和 Binary 方式中，如果没有 Access 选项，Open 语句将按下列顺序打开文件： Read Write → Read → Write

- ④ <锁定类型> 用来在多用户或多进程环境中，限定其他用户或进程打开文件的操作，其取值见表 14-3。

表 14-3 锁定类型

参 数	说 明	参 数	说 明
Shared	与其他进程共享打开的文件	Lock Write	不允许其他进程写该文件
Lock Read	不允许其他进程读该文件	Lock Read Write	不允许其他进程读/写该文件

如果不指定〈锁定类型〉,则在文件打开时,不允许其他进程访问该文件。

⑤〈文件号〉是打开文件时指定的文件句柄,取值是 1~511 之间的整数。**Open** 语句将打开的文件与指定的文件号联系在一起,在文件的读/写操作中,以文件号代替文件名。使用 **FreeFile** 函数可得到下一个可用的文件号。

⑥〈记录长度〉为一个不超过 32 767 的整数。对于随机文件,该值表示记录长度;对于顺序文件,该值代表缓冲字符数;对于二进制文件,Len 选项被忽略。

例如:

**Open "c:\WEXAM\student.dat" For Output As #1**

表示如果文件“c:\WEXAM\student.dat”不存在,就用此文件名建立一个新文件;如果该文件存在,在打开文件时,文件中原有数据全部丢失,新记录将从头开始写入。因此,无论文件存在与否,参数 **Output** 都将建立一个新文件。

在使用 **Open** 语句时,应注意以下 3 点。

① **Open** 语句有打开和建立两种功能,如果语句中指定的文件不存在,则在使用 **Output**, **Append**, **Random** 或 **Binary** 方式时,将以指定的文件名建立一个新文件。

② 为了满足多种存取数据的需求,在 **Input**, **Random** 和 **Binary** 方式下,不必关闭文件,就可用不同的文件号打开同一文件。而在 **Append** 和 **Output** 方式下,则必须先关闭文件,才可用不同的文件号打开同一文件,否则将产生“文件已打开”的错误。

③ 如果文件已由其他进程打开,而且指定了不允许的访问类型,则 **Open** 操作失败。

## 2. Close 语句

**Close** 语句用来关闭 **Open** 语句所打开的输入/输出文件。其语法格式为:

**Close** [**#**〈文件号〉] [, **#**〈文件号〉] ...

### 【说明】

①〈文件号〉为欲关闭文件的文件号。省略时,关闭 **Open** 语句打开的所有文件。

② 执行 **Close** 语句,文件与其〈文件号〉之间的关联将终结,并释放与该文件相关联的缓冲区空间。

③ 当关闭用参数 **Output** 或 **Append** 打开的文件时,系统将把文件缓冲区中的数据写入文件中。

④ 执行 **End** 语句和 **Reset** 语句也会关闭所有用 **Open** 语句打开的磁盘文件。

例如:

<b>Close #1,#3</b>	' 关闭#1, #3 文件
<b>Close</b>	' 关闭所有打开的文件

## 14.2.3 文件访问函数

### 1. EOF 函数

**EOF** 函数用于测试指定文件的结束状态,通常用来检查以 **Input** 方式打开的顺序文件。其语法格式为:

**EOF**(〈文件号〉)

【说明】当文件中的记录指针指向文件末尾时(最后一条记录的后面),**EOF** 函数返回 **True**,否则返回 **False**。如果在文件末尾执行输入操作,VB 将给出错误信息“输入超出文件

尾”。使用 EOF 可以避免这种错误的发生。

## 2. FreeFile 函数

FreeFile 函数返回指定范围内下一个可用的文件号。其语法格式为：

**FreeFile[(〈区间号〉)]**

【说明】〈区间号〉为 0 或省略时，返回 1~255 之间的文件号；若〈区间号〉为 1，则返回 256~511 之间的文件号。

使用 FreeFile 函数可以把一个未使用的文件号赋给指定变量，当用 Open 语句打开文件时，使用代表文件号的变量，可以不必考虑具体的文件号。当打开的文件较多，尤其是在一些通用过程中访问文件时，可以避免打开正在使用的文件号。例如：

```
FileNumber = FreeFile
FileName = "c:\WEXAM\faz2001.dat"
Open FileName For Input As #FileNumber
.....
Close #FileNumber
```

## 3. Input 函数

Input 函数返回它所读出的所有字符，包括逗号、空格符、引号及回车符和换行符等，可用于以 Input 方式打开的顺序文件或以二进制文件方式打开的文件。其语法格式为：

**Input(〈字符个数〉, [#]〈文件号〉)**

【说明】〈字符个数〉指定了需要返回的字符个数。

## 4. Len 函数

Len 函数返回字符串表达式中包含字符的数目，或存储一个变量所需的字节数。其语法格式为：

**Len(〈字符串表达式〉)**

【说明】对于用户定义类型，如一个记录类型变量，Len 函数返回的是该变量写入文件时的大小。

VB 提供的另一个测试字符串长度的函数是 LenB。与 Len 函数不同的是，该函数返回的是代表字符串的字节数，而不是字符串中字符的数目。当字符串中含有汉字时，应该特别注意区分，例如：

```
MyLen = Len("计算机等级考试")           ' 返回 7
MyLen = LenB("计算机等级考试")           ' 返回 14
```

对于定长字符串，Len 函数总是返回字符串定义的长度，例如：

```
Dim MyStr As String * 20
MyStr = "计算机等级考试"                  ' 实际输入的字符个数为 7
MyLen = Len(MyStr)                        ' 返回变量 MyStr 定义的长度为 20
```

## 5. Loc 函数

Loc 函数返回一个用 Open 语句打开的文件的上一次读/写的位置。其语法格式为：

**Loc(〈文件号〉)**

【说明】对于随机文件，Loc 返回上一次对文件进行读出或写入操作的记录号；对于二进制文件，Loc 返回上一次读出或写入的字节位置；对于顺序文件，由于文件只能从头到尾顺序写入或读出，因此，一般不需要使用 Loc 函数。

随机文件的记录号从 1 开始，文件字节的起始位置也是 1。

## 6. LOF 函数

LOF 函数返回用 Open 语句打开的文件的大小（以字节为单位）。其语法格式为：

**LOF(〈文件号〉)**

【说明】要取得一个尚未打开的文件的大小，应该使用 FileLen 函数。

## 7. Seek 函数

Seek 函数返回一个用 Open 语句打开的文件的当前读/写位置。其语法格式为：

**Seek(〈文件号〉)**

【说明】对于随机文件，Seek 返回文件指针指向的将要读出或写入的记录号；对于二进制文件和顺序文件，Seek 返回将要读出或写入的字节位置。

# 14.3 顺序文件的操作

顺序文件的打开与关闭操作由 Open 语句和 Close 语句来实现。打开顺序文件的基本语法格式为：

**Open 〈文件名〉 For {Input | Output | Append} As 〈文件号〉 [Len = buffersize]**

其中，参数说明参见表 14-1。

下面介绍用来对顺序文件进行读/写操作的语句。

### 14.3.1 顺序文件的写操作

要将数据写入文本文件，应以 Output 或 Append 方式打开该文件，然后使用 Print #或者 Write #语句将数据写入文件中。

#### 1. Print #语句

Print #语句将格式化数据写入顺序文件中，其功能与多次使用 Print 方法类似，只是 Print 方法输出的对象是窗体、图片框或打印机，而 Print # 语句的输出对象是文件。其语法格式为：

**Print # 〈文件号〉, [{Spc(n) | Tab(n)}] [〈表达式列表〉] [{, | ; }]**

【说明】

① 〈文件号〉是打开文件时指定的文件句柄。

② Spc 函数、Tab 函数、〈表达式列表〉及尾部的逗号、分号等与 Print 方法中用法相同。

③ 若省略〈文件号〉后各项，则向文件中输出一个空行。

④ 如果需要读出 Print # 语句在文件中写入的数据，可使用 Line Input # 语句或 Input 函数。

【例 14-1】 用 Print #语句向顺序文件输出数据。

编写窗体的 Click 事件代码：

```
Private Sub Form_Click()  
    Open "c:\WEXAM\26160001\out1.txt" For Output As #1  
    Print #1, 1; 2; 3; 4; 5                                ' 用紧凑格式输出数值型数据  
    Print #1, "计算机"; "等级考试"; "1"; "2"; "3"        ' 用紧凑格式输出字符型数据  
    Print #1, "4", "5", 12.35, 12 - 76                    ' 用标准格式输出  
    Print #1,                                              ' 输出一个空行  
    Print #1, "这是用"; "Print # 语句";                  ' 注意，输出列表最后有分号  
    Print #1, "输出的文件"                                ' 紧凑上一个 Print # 语句输出  
    Close #1  
End Sub
```

运行程序，在窗体上单击后，用 Windows 的“记事本”打开文件 out1.txt，可以看到文件的内容及其格式，如图 14-1 所示。



图 14-1 用 Print #语句写入文件的格式示例

2. Write #语句

与 Print #语句相同，Write # 语句将输出列表指定的数据，顺序写入〈文件号〉所代表的文件中。其语法格式为：

Write # 〈文件号〉, [ 〈表达式列表〉 ]

其中，〈文件号〉的含义同上；〈表达式列表〉中若有多个表达式，表达式之间用逗号隔开，也可用空格或分号分隔；如果省略〈表达式列表〉，则输出一个空白行到文件中。若要从文件中读出用 Write #语句写入的数据，通常使用 Input # 语句。

用 Write # 语句写入的顺序文件具有如下的格式：

- ① 字符型数据，用双引号 “ ” 括起来。
- ② 逻辑型数据，保存为 #TRUE# 或者 #FALSE#。
- ③ 日期型数据，采用 #yyyy-mm-dd hh:mm:ss# 的格式，或将日期部分和时间部分分开处理，其形式为 #yyyy-mm-dd# 和 #hh:mm:ss# 。
- ④ 各数据项之间以紧凑格式存放，并自动插入分界符（逗号）。
- ⑤ 在输出列表中的最后一个字符被写入文件后，自动插入回车换行符。

【例 14-2】 用 Write # 语句向顺序文件输出数据。

编写窗体的 Click 事件代码：

```
Private Sub Form_Click()  
    Dim I As Integer, S As String
```

```

Dim D As Date, B As Boolean
Open "c:\WEXAM\26160001\out2.txt" For Output As #1    ' 以顺序输出方式建立并打开文件
I = 100: S = "773456"
D = Date: B = True                                     ' Date 函数返回系统当前的日期
Write #1, I, S, D, B                                   ' 写入第一条记录
Write #1, "这是用 Write 语句输出的文件"              ' 写入第二条记录
Close #1                                                ' 关闭文件

```

### End Sub

运行程序，在窗体上单击后，用 Windows 的“记事本”打开文件 out2.txt，可以看到文件的内容及其格式，如图 14-2 所示。此例可以帮助我们理解 Write # 语句的具体使用方法，以及文件中各种类型数据的存放格式。



图 14-2 用 Write # 语句写入文件的格式示例

Print # 语句与 Write # 语句的区别在于：Print # 语句根据指定的格式，将数据写入文件，数据项之间不会自动插入分界符；而 Write # 语句在数据项之间自动插入分界符。

Write # 语句适合输出不同类型的数据，特别是当数据写入文件后，还需用其他程序读出进行处理的情况；Print # 语句适合输出文本类型或列表格式的数据，供打印或显示用。

**【例 14-3】** 用 Write # 语句向顺序文件输出 20 个随机两位整数。

编写窗体的 Click 事件代码：

```

Private Sub Form_Click()
    Randomize
    Open "c:\WEXAM\26160002\datain.txt" For Output As #1
    For i = 1 To 20
        Write #1, Int(Rnd * 90) + 10
    Next
    Close #1

```

### End Sub

**【例 14-4】** 设在工程中有一个标准模块，其中定义了如下记录类型(用户自定义类型)：

### Type Books

```

Name As String * 10
TelNum As String * 20

```

### End Type

执行下列事件过程代码将在顺序文件中写入一条记录：

```

Private Sub Command1_Click()
    Dim b As Books

```

```

Open "c:\WEXAM\Person.txt" For Output As #1
b.Name = InputBox("输入姓名")
b.TelNum = InputBox("输入电话号码")
Write #1, b.Name, b.TelNum
Close #1
End Sub

```

### 14.3.2 顺序文件的读操作

要读取文本文件的内容，应以 **Input** 方式打开该文件，然后使用 **Input #** 或者 **Line Input #** 语句将文件复制到内存变量中。

#### 1. Input # 语句

**Input** 语句从一个打开的顺序文件中读出数据，并将数据赋给指定的变量，其语法格式为：

**Input #** 〈文件号〉, 〈变量列表〉

##### 【说明】

- ① 〈文件号〉为打开文件时指定的文件句柄。
- ② 〈变量列表〉为接收数据的变量，变量之间用逗号分隔。**Input #** 语句把从文件中读出的数据赋给这些变量，变量的个数和类型应该与文件中读取的数据的个数和类型一致。
- ③ 用 **Input #** 语句把读出的数据赋给数值变量时，将忽略前导空格、回车符或换行符，把遇到的第一个非空格、非回车符或非换行符作为数值的开始，再遇到空格、回车符或换行符则认为数值结束。对于字符串数据，同样忽略前导空格、回车符或换行符，如果需要把开头带有空格的字符串赋给变量，则必须把字符串放在双引号中。

通常，用 **Write #** 语句写入文件的数据，可使用 **Input #** 语句读出。读数据时，一般不需要处理就可直接将数据指定给变量。

**【例 14-5】** 用 **Input #** 语句读取例 14-2 所建立文件中的数据，并将数据显示在窗体上。编写窗体的 Click 事件代码：

```

Private Sub Form_Click()
    Dim I As Integer, S As String
    Dim D As Date, B As Boolean
    Open "c:\WEXAM\26160001\out2.txt" For Input As #1      ' 以顺序输入方式打开文件
    Input #1, I, S, D, B                                   ' 读取第一条记录
    Close #1                                               ' 关闭文件
    Cls
    Print "文件中第一条记录为： "
    Print I, S, D, B
End Sub

```

程序运行结果如图 14-3 所示。

注意，在用 **Input #** 语句读取用 **Write #** 语句写入文件的数据时，并没有对字符串数据的



定界符“”做特别的处理，也没有删除日期型数据和布尔型数据的定界符“#”。Input # 语句在将数据指定给变量时，会自动去掉这些定界符，如同 Write # 语句在写文件时会根据变量的类型自动添加定界符一样。

【例 14-6】 创建数据文件 datain1.txt 和 datain2.txt，方法同例 14-3。设计程序分别读入文件中的各 20 个整数，分别放入两个数组 Arr1 和 Arr2 中，然后把两个数组中对应下标的元素相加，其结果放入第三个数组中（即，第一个数组的第  $n$  个元素与第二个数组的第  $n$  个元素相加，其结果作为第三个数组的第  $n$  个元素， $n=1, 2, \cdots, 20$ ），并计算第三个数组各元素之和，最后把所求得的和在窗体上显示出来，并将计算结果存入 dataout.txt 文件中。

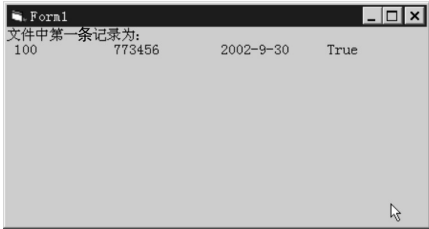


图 14-3 用 Input # 语句读取文件

设计步骤如下。

（1）设计程序界面并设置控件属性。

在窗体中增加 3 个命令按钮 Command1~Command3，并修改其属性，参见表 14-4。

表 14-4 属性设置

对 象	属 性	属 性 值
Command1	Caption	读入数据
	Name	C1
Command2	Caption	计算
	Name	C2
Command3	Caption	存盘
	Name	C3

（2）编写代码。

首先在窗体的通用段声明数组及变量：

```
Option Base 1
Dim Arr1(20) As Integer
Dim Arr2(20) As Integer
Dim s As Integer
```

然后编写读取和写入数据的通用过程。

- 读取文件 datain1.txt 的 ReadDate1 过程代码如下：

```
Sub ReadData1()
    Open App.Path & "\ " & "datain1.txt" For Input As #1
    For i = 1 To 20
        Input #1, Arr1(i)
    Next i
    Close #1
End Sub
```

- 读取文件 datain2.txt 的 ReadDate2 过程代码如下：

**Sub ReadData2()**

```
Open App.Path & "\" & "datain2.txt" For Input As #1
For i = 1 To 20
    Input #1, Arr2(i)
Next i
Close #1
```

**End Sub**

- 写入文件的 WriteData 过程代码如下：

**Sub WriteData(Filename As String, Num As Integer)**

```
Open App.Path & "\" & Filename For Output As #1
Print #1, Num
Close #1
```

**End Sub**

其中，App.Path 表示当前工程所在的文件夹（目录）。  
最后编写事件过程代码。

- “读入数据”命令按钮 C1 的 Click 事件代码如下：

**Private Sub C1\_Click()**

```
Call ReadData1
Call ReadData2
```

**End Sub**

- “计算”命令按钮 C2 的 Click 事件代码如下：

**Private Sub C2\_Click()**

```
Dim Arr3(20) As Integer
For i = 1 To 20
    Arr3(i) = Arr1(i) + Arr2(i)
Next
s = 0
For i = 1 To 20
    s = s + Arr3(i)
Next
Cls
Print
Print "各元素之和为：", s
```

**End Sub**

- “存盘”命令按钮 C3 的 Click 事件代码如下：

**Private Sub C3\_Click()**

```
WriteData "dataout.txt", s
```

**End Sub**

运行程序，依次单击“读入数据”、“计算”、“存盘”命令按钮（如图 14-4 所示）后，所得结果保存在当前文件夹的 dataout.txt 文件中。

## 2. Line Input #语句

Line Input #语句从顺序文件中读出一行，并把它赋给一个字符串变量，其语法格式为：

**Line Input # 〈文件号〉, 〈字符串变量〉**

其中，〈文件号〉的含义同上，〈字符串变量〉用来接收从文件中读出一行字符。

Line Input # 语句通常用来读取用 Print # 语句写入的文件。它一次从文件中读取一行字符，直到遇到回车符为止。回车符将被忽略，不会附加到字符串上。Line Input # 语句也可用来读取以 ASCII 码存放在磁盘上的各种源程序文件及文本文件等。

**【例 14-7】** 设在当前目录中已有顺序文件 in1.txt，文件中有几行汉字，如图 14-5 所示。在窗体上画一个文本框，名称为 Text1，能显示多行；再画一个命令按钮，名称为 C1，标题为“存盘”。编写适当的事件过程，使得在加载窗体时，把 in1.txt 文件的内容显示在文本框中，然后在文本的最前面手工插入一行汉字：“计算机等级考试”。最后单击“存盘”按钮，可以把在文本框中修改过的内容存到文件 out1.txt 中，如图 14-6 所示。

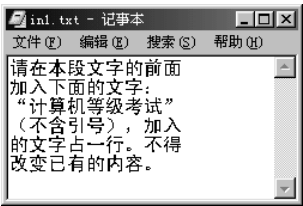


图 14-5 顺序文件 in1.txt

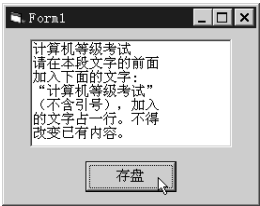


图 14-6 写入文本

窗体的设计如图 14-6 所示，下面给出程序代码。

窗体的 Load 事件代码可以读入文件 in1.txt：

```
Private Sub Form_Load()  
    Open App.Path & "\" & "in1.txt" For Input As #1  
    a = ""  
    Do Until EOF(1)  
        Line Input #1, b  
        a = a & b & Chr(13) & Chr(10)  
    Loop  
    Close #1  
    Text1.Text = a  
End Sub
```

命令按钮 C1 的 Click 事件代码可以将文本框中的内容写入文件 Out1.txt 中：

```
Private Sub C1_Click()  
    Open App.Path & "\" & "out1.txt" For Output As #1  
    Print #1, Text1.Text  
    Close #1  
End Sub
```

【例 14-8】 一个简易文本编辑器，具有创建、编辑、保存普通文本文件的功能，如图 14-7 所示。

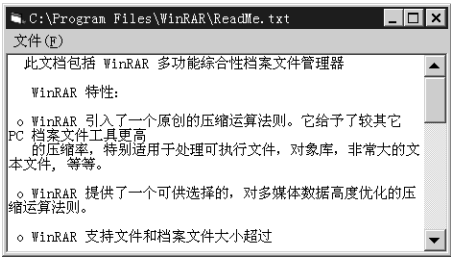


图 14-7 简易的文本编辑器

设计步骤如下。

(1) 建立应用程序用户界面并设置对象属性。

首先在窗体上添加一个文本框 Text1 和一个公共对话框 CommonDialog1，然后打开菜单编辑器，按照表 14-5 设计菜单项。

表 14-5 菜单项的设置

标题 (Caption)	名称 (Name)	索引 (Index)	说 明
文件(&F)	Files		主菜单项 1
....新建(&N)	File	0	子菜单项 11
....打开(&O)	File	1	子菜单项 12
....保存(&S)	File	2	子菜单项 13
....另存(&A)	File	3	子菜单项 14
....关闭(&X)	File	4	子菜单项 15

窗体、文本框、公共对话框的属性设置参见表 14-6。

表 14-6 属性设置

对 象	属 性	属 性 值	说 明
Form1	Caption	未命名	窗体的标题
CommonDialog1	Filter	所有文件(*.*)*. * 文本文件(*.TXT)*.txt	文件过滤器
	FilterIndex	2	文件过滤器索引指向第二项
Text1	Text		文本框的内容
	MultiLine	True	显示多行文本
	ScrollBars	2 - Vertical	垂直滚动条

(2) 编写代码。

为了使改变窗体大小的时候文本框能随之改变，编写窗体的 Resize 事件代码：

```
Private Sub Form_Resize()  
    Text1.Left = 0  
    Text1.Top = 0  
    Text1.Height = Form1.ScaleHeight  
    Text1.Width = Form1.ScaleWidth - Picture1.Width  
End Sub
```

编写菜单按键数组 File() 的 Click 事件代码:

**Private Sub File\_Click(Index As Integer)**

n = Index

Select Case n

Case 0 ' 新建

Text1.Text = ""

Form1.Caption = "未命名"

Case 1 ' 打开

CommonDialog1.ShowOpen ' 显示“打开”公共对话框

fname = CommonDialog1.FileName

If fname <> "" Then

Text1.Text = ""

Open fname For Input As #1

b = ""

Do Until EOF(1)

Line Input #1, nextline

b = b & nextline & Chr(13) & Chr(10)

Loop

Close #1

Text1.Text = b

End If

Form1.Caption = fname

Case 2 ' 保存

If Form1.Caption = "未命名" Or Form1.Caption = "" Then

CommonDialog1.ShowSave '显示“另存为”公共对话框

fname = CommonDialog1.FileName

Else

fname = Form1.Caption

End If

If fname <> "" Then

Open fname For Output As #1

Print #1, Text1.Text

Close #1

End If

Case 3 ' 另存

CommonDialog1.ShowSave ' 显示“另存为”公共对话框

fname = CommonDialog1.FileName

If fname <> "" Then

Open fname For Output As #1

```

        Print #1, Text1.Text
    Close #1
End If
Case 4
    Text1.Text = ""
End
End Select
Text1.SetFocus
End Sub

```

## 14.4 随机文件的操作

随机文件中的一行数据称为一条记录。随机文件对文件的读/写顺序没有限制，可以随意读/写某一条记录。这就要求记录的长度是固定的，以便由记录号来定位。随机文件的读/写速度较快，但其占用空间较大。

随机文件的打开与关闭操作仍由 **Open** 语句和 **Close** 语句来实现。打开随机文件的基本语法格式为：

**Open** 〈文件名〉 [**For Random**] **As** 〈文件号〉 **Len =** 〈记录长度〉

下面介绍用来对随机文件进行读/写操作的语句。

### 14.4.1 随机文件的读/写操作

对打开的随机文件中的记录进行编辑，要先把记录从文件读到内存变量中，然后就可以修改变量的值了，最后再把变量写回该文件。

#### 1. Get #语句

使用 **Get #** 语句把记录读入变量，其语法格式为：

**Get #** 〈文件号〉, 〈记录号〉, 〈变量名〉

其中，〈文件号〉是打开文件时指定的文件句柄，〈记录号〉是要读入的记录号数，而〈变量名〉则是接收记录内容的记录型变量名，一般声明为用户定义类型。

#### 2. Put #语句

使用 **Put #** 语句可以把数据写入或替换随机文件中的记录，其语法格式为：

**Put #** 〈文件号〉, 〈记录号〉, 〈变量名〉

其中，〈文件号〉是打开文件时指定的文件句柄，〈记录号〉是要写入或替换的记录位置，〈变量名〉是接收记录的内容的记录型变量名。

**【例 14-9】** 利用随机文件保存学生的成绩，可以浏览或编辑（读取、修改、保存）学生的学号、姓名及 3 门功课的成绩，界面如图 14-8 所示。

设计步骤如下。

（1）建立应用程序用户界面并设置对象属性。首先在窗体中增加两个框架 **Frame1**，

Frame2 和 3 个命令按钮 Command1~Command3。然后选定 Frame1，在其中增加一个文本框数组 Text1(0)~Text1(4)和 5 个标签；选定 Frame2，在其中增加一个列表框 List1。各框架、文本框、标签、列表框和命令按钮的属性设置如图 14-8 所示。



图 14-8 简易的学生成绩管理软件

(2) 编写代码。

首先在窗体的通用段创建用户定义类型（记录类型）并声明变量：

```
Private Type cj
    xm As String * 6
    xh As String * 6
    sx As Integer
    yw As Integer
    wy As Integer
```

```
End Type
```

```
Private da As cj
```

编写“读取”按钮的 Click 事件代码：

```
Private Sub Command1_Click()
    Dim sx As Single, yw As Single, wy As Single
    FileName = App.Path & "\" & "xsda2.dat"
    Open FileName For Random As #1 Len = Len(da) ' 打开随机数据文件
    lastrec = LOF(1) / Len(da)
    List1.Clear
    For n = 1 To lastrec
        Get #1, n, da
        xh = Format(da.xh, "@@@@@@")
        xm = Format(RTrim(da.xm), "@@@@")
        yw = Format(da.yw, "####")
        wy = Format(da.wy, "####")
        sx = Format(da.sx, "####")
        msg = xh & " " & yw & " " & wy & " " & sx & " " & xm
        List1.AddItem msg
    Next
```

```
Command1.Enabled = False
```

**End Sub**

编写“保存”按钮的 Click 事件代码:

**Private Sub Command2\_Click()**

```
recnum = List1.ListIndex
da.xh = Text1(0).Text
da.xm = Text1(1).Text
da.yw = Text1(2).Text
da.wy = Text1(3).Text
da.sx = Text1(4).Text
xh = Format(da.xh, "@@@@@@")
xm = Format(RTrim(da.xm), "@@@@")
yw = Format(da.yw, "####")
wy = Format(da.wy, "####")
sx = Format(da.sx, "####")
msg = xh & " " & yw & " " & wy & " " & sx & " " & xm
Put #1, recnum + 1, da
List1.RemoveItem recnum
List1.AddItem msg, recnum
Command2.Enabled = False
```

**End Sub**

编写“关闭”按钮的 Click 事件代码:

**Private Sub Command3\_Click()**

```
Close #1
Unload Me
```

**End Sub**

编写列表框的 Click 事件代码:

**Private Sub List1\_Click()**

```
If List1.ListIndex > -1 Then
    n = List1.ListIndex + 1
    Get #1, n, da
    Text1(0).Text = da.xh
    Text1(1).Text = da.xm
    Text1(2).Text = da.yw
    Text1(3).Text = da.wy
    Text1(4).Text = da.sx
End If
Command2.Enabled = True
```

**End Sub**



## 14.4.2 随机文件中记录的增加与删除

### 1. 增加记录

在随机文件中添加记录，是指向文件的末尾添加记录。其方法是：把〈记录号〉的值设置为比文件中的记录数多 1，然后使用 Put 语句。

【例 14-10】 在例14-9中增加一个添加新记录的功能，如图 14-9 所示。

只需在例14-9中增加一个“添加”命令按钮 Command4，并编写其 Click 事件代码：

```
Private Sub Command4_Click()  
    lastrec = LOF(1) / Len(da) + 1  
    da.xh = Text1(0).Text  
    da.xh = Text1(0).Text  
    da.xm = Text1(1).Text  
    da.yw = Text1(2).Text  
    da.wy = Text1(3).Text  
    da.sx = Text1(4).Text  
    xh = Format(da.xh, "@@@@@@")  
    xm = Format(RTrim(da.xm), "@@@@@")  
    yw = Format(da.yw, "#####")  
    wy = Format(da.wy, "#####")  
    sx = Format(da.sx, "#####")  
    msg = xh & " " & yw & " " & wy & " " & sx & " " & xm  
    Put #1, lastrec, da  
    List1.AddItem msg  
    Command2.Enabled = False
```

End Sub



图 14-9 增加一个添加新记录的功能

### 2. 删除记录

通过清除其字段可以删除一个记录，但是该记录仍在文件中存在。通常文件中不能有空记录，因为它们会浪费空间且会干扰顺序操作，最好把余下的记录复制到一个新文件中，然后删除老文件。



图 14-10 增加一个删除记录的功能

要清除随机访问文件中删除的记录，步骤如下。

- (1) 创建一个新文件。
- (2) 把有用的所有记录从原文件复制到新文件中。
- (3) 关闭原文件并用 Kill 语句删除它（语法参见

14.6.2 节）。

- (4) 使用 Name 语句把新文件以原文件的名字重新命名（语法参见 14.6.2 节）。

【例 14-11】 在例 14-10 中增加一个删除记录的功能，如图 14-10 所示。

只需在例 14-10 中增加一个“删除记录”命令按钮 Command5，其 Click 事件代码如下：

```
Private Sub Command5_Click()  
    FileName = App.Path & "\" & "xsda2.dat"  
    recnum = List1.ListIndex + 1
```

```

Open "rec.tem" For Random As #2 Len = Len(da)      ' 打开临时随机文件
lastrec = LOF(1) / Len(da)
For n = 1 To lastrec
    If n <> recnum Then
        Get #1, n, da
        Put #2, , da
    Else
        Get #2, n, da
        With da
            Text1(0).Text = .xh
            Text1(1).Text = .xm
            Text1(2).Text = .yw
            Text1(3).Text = .wy
            Text1(4).Text = .sx
        End With
    End If
Next
Close #1
Close #2
Kill FileName
Name "rec.tem" As FileName
Call Command1_Click
End Sub

```

## 14.5 文件系统控件

文件系统控件的作用是显示关于磁盘驱动器、目录和文件的信息，并从中进行选择以便执行进一步的操作。通过使用驱动器列表框（DriveListBox）、目录列表框（DirListBox）和文件列表框（FileListBox）这 3 种控件的组合，可以创建自定义文件系统对话框。

### 14.5.1 驱动器列表框

驱动器列表框的外观与组合框相似，提供一个下拉式驱动器清单，可以显示当前系统中所有有效的磁盘驱动器。

#### 1. 驱动器列表框的属性

驱动器列表框最主要的属性是 Drive 属性，该属性用于设置或返回要操作的驱动器。该属性只能在运行时由程序代码设置或访问，设计阶段无效。

例如，若驱动器列表框的对象名为 Drive1，要获得当前驱动器号，可使用如下代码：

```
Drivename = Drive1.Drive
```

若要设置当前驱动器为 D 盘，则可使用如下代码：

```
Drive1.Drive = "D:"
```

另外，使用 ChDrive 语句也可以将指定的驱动器设为当前驱动器，如：

```
ChDrive Drive1.Drive ' 将用户在列表框中选择的驱动器设为当前驱动器
```

```
ChDrive "D" ' 将 D 盘设为当前驱动器
```

驱动器列表框常与目录列表框和文件列表框配合使用，以完成对相关文件的控制。

## 2. 驱动器列表框的事件

驱动器列表框的常用事件主要是 Change 事件，该事件在驱动器列表框的 Drive 属性值发生改变时产生。通常在该事件过程中编程，以完成相关的操作。

### 14.5.2 目录列表框

目录列表框用于显示当前驱动器或指定驱动器上的目录结构。显示时从根目录开始，各级子目录按目录的层次结构依次缩进。

#### 1. 目录列表框的属性

目录列表框最主要的属性是 Path 属性，用于设置或返回要显示目录结构的驱动器路径或目录路径。该属性仅在运行时有效，设计时无效。在程序中，通过访问该属性，可获得列表框中显示的当前目录。例如，若目录列表框的对象名为 Dir1，要获得当前工作目录，可使用如下代码：

```
Dir1.Path = Drive1.Drive
```

要设置当前目录为 C:\Windows，可使用如下代码：

```
Dir1.Path = "C:\Windows"
```

使用 ChDir 语句也可以改变当前的目录或文件夹，如：

```
ChDir Dir1.Path ' 将用户在目录列表框中选取的目录设为当前目录
```

#### 2. 目录列表框的事件

目录列表框能响应一些常用的事件，在实际编程中，最常用的是 Change 事件。该事件在目录列表框的 Path 属性发生改变时产生。

目录列表框常与驱动器列表框配合使用，以便在驱动器改变时，目录列表框中的显示也能跟着改变。实现的方法通常是在驱动器的 Change 事件中为目录列表框的 Path 属性赋值。

### 14.5.3 文件列表框

文件列表框常与目录列表框配合使用，用以显示指定目录下的文件列表。用户可从文件列表框中选择所要操作的一个或多个文件。

#### 1. 文件列表框的属性

文件列表框可视为标准列表框的一个衍生列表框或具体化，因此，除了自身特有的属性外，它也继承了标准列表框的一些重要属性。其主要属性见表 14-7。

表 14-7 FileListBox 的属性

属 性	说 明
FileName	返回或设置所选文件的路径和文件名，设计时不可用
MultiSelect	是否允许用户选择多个文件，True：允许，False：不允许
Path	设置或返回要显示文件列表框的文件路径
Pattern	设定允许显示文件名的文件类型，如*.exe;*.com 等，默认值为*.*
Archive	是否可以显示 Archive 属性的文件
Hidden	是否可以显示 Hidden 属性的文件
Normal	是否可以显示 Normal 属性的文件
ReadOnly	是否可以显示 ReadOnly 属性的文件
System	是否可以显示 System 属性的文件

## 2. 文件列表框的事件

文件列表框常用的事件主要有 PathChange, PatternChange, DblClick, Click, GotFocus 和 LostFocus。

### 14.5.4 文件系统控件共有的属性

以下 3 个属性是文件系统控件共有的：

- ListCount 属性。该属性用于返回列表框中列表项总数，仅在运行时有效。
- ListIndex 属性。该属性用于设置或返回列表框中所选定列表项的索引值，与在标准列表框中的功能和用法相同，仅在运行时有效。
- List 属性。该属性为数组性质的属性，用于设置或返回文件列表框中指定列表项的内容，与在标准列表框中的功能和用法相同，仅在运行时有效。

### 14.5.5 文件系统对象的同步操作

直接绘制在窗体中的驱动器列表框、目录列表框和文件列表框，彼此之间并无任何联系，为了使它们能同步操作，就需要通过编程，将它们彼此关联起来。一般，可以在 Drive1\_Change 事件和 Dir1\_Change 事件中分别加入语句：Dir1.Path = Drive1.Drive 和 File1.Path = Dir1.Path。

**【例 14-12】** 使用文件系统控件制作简易的文本浏览器，界面如图 14-11 所示。设计步骤如下。

（1）建立应用程序用户界面并设置对象属性。

首先在窗体中增加一个框架 Frame1。选定 Frame1，在其中增加一个驱动器列表框 Drive1、一个目录列表框 Dir1、一个文件列表框 File1 和一个文本框 Text1。

文件系统控件的属性设置参见表 14-8。

框架、文本框的属性设置参见图 14-11。

表 14-8 文件系统控件的属性设置

对 象	属 性	属 性 值	说 明
File1	Pattern	*.txt	允许显示的文件类型

(2) 编写代码。

编写驱动器列表框 Drive1 的 Change 事件代码：

```
Private Sub Drive1_Change()  
    Dir1.Path = Drive1.Drive  
  
End Sub
```

编写目录列表框 Dir1 的 Change 事件代码：

```
Private Sub Dir1_Change()  
    File1.Path = Dir1.Path  
  
End Sub
```

编写文件表框 File1 的 Click 事件代码：

```
Private Sub File1_Click()  
    ChDrive Drive1.Drive  
    ChDir Dir1.Path  
    Text1.Text = ""  
    Open File1.FileName For Input As #1  
    b = ""  
    Do Until EOF(1)  
        Line Input #1, nextline  
        b = b & nextline & Chr(13) & Chr(10)  
    Loop  
    Close #1  
    Text1.Text = b  
  
End Sub
```

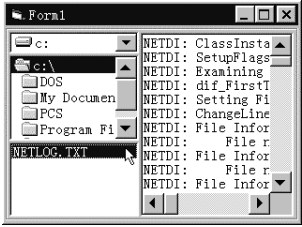


图 14-11 简易的文本浏览器

## 14.6 文件基本操作

文件的基本操作是指对文件的删除、复制、移动、更名等操作。VB 不仅提供了对文件操作的相应命令，还提供了对目录（文件夹）操作的相应命令。

### 14.6.1 目录的基本操作

#### 1. Mkdir 语句

使用 Mkdir 语句可以创建一个新的目录或文件夹。其语法格式为：

```
Mkdir <路径名>
```

其中，<路径名>是用来指定所要创建的目录或文件夹的字符串表达式，可以包含驱动器。如果没有指定驱动器，则 Mkdir 会在当前驱动器中创建新的目录或文件夹。

#### 2. ChDir 语句

使用 ChDir 语句可以改变当前的目录或文件夹。其语法格式为：

```
ChDir <路径名>
```

其中,〈路径名〉是一个字符串表达式,它指明哪个目录或文件夹将成为新的默认目录或文件夹。〈路径名〉中可以包含驱动器名。如果没有指定驱动器,则 ChDir 将在当前的驱动器上改变默认目录或文件夹。

【说明】ChDir 语句改变默认目录位置,但不会改变默认驱动器位置。例如,如果默认的驱动器是 C,则下面的语句将会改变驱动器 D 上的默认目录,但是 C 仍然是默认的驱动器:

```
ChDir "D:\TMP"
```

### 3. Rmdir 语句

使用 Rmdir 语句可以删除一个存在的目录或文件夹。其语法格式为:

```
Rmdir 〈路径名〉
```

其中,〈路径名〉是一个字符串表达式,用来指定要删除的目录或文件夹。〈路径名〉中可以包含驱动器名。如果没有指定驱动器,则 Rmdir 会在当前驱动器上删除目录或文件夹。

【说明】如果想要使用 Rmdir 来删除一个含有文件的目录或文件夹,则会发生错误。在试图删除目录或文件夹之前,应先使用 Kill 语句来删除所有文件。

### 4. CurDir 函数

CurDir 函数返回一个表示当前路径的字符串。其语法格式为:

```
CurDir [(〈驱动器名〉)]
```

其中,可选的〈驱动器名〉参数是一个字符串表达式,它指定一个存在的驱动器。如果没有指定驱动器,或者〈驱动器名〉是零长度字符串(""),则 CurDir 会返回当前驱动器的路径。

### 5. ChDrive 语句

使用 ChDrive 语句可以改变当前的驱动器。其语法格式为:

```
ChDrive 〈驱动器名〉
```

其中,〈驱动器名〉是一个字符串表达式,它指定一个存在的驱动器。如果使用零长度的字符串(""),则当前的驱动器将不会改变。如果〈驱动器名〉参数中有多个字符,则 ChDrive 只会使用首字母。

## 14.6.2 文件的基本操作

### 1. Kill 语句

使用 Kill 语句可以从磁盘中删除文件。其语法格式为:

```
Kill 〈文件名〉
```

其中,〈文件名〉是用来指定文件名的字符串表达式,可以包含目录或文件夹及驱动器。

【说明】Kill 语句支持多字符(\*)和单字符(?)的通配符来指定多重文件。

### 2. Name 语句

使用 Name 语句可以重新命名一个文件、目录或文件夹。其语法格式为:

```
Name 〈旧文件名〉 As 〈新文件名〉
```

其中,〈旧文件名〉为字符串表达式,指定已存在的文件名和位置,可以包含目录或文件夹及驱动器。〈新文件名〉也是字符串表达式,指定新的文件名和位置,可以包含目录或文件夹及驱动器。由〈新文件名〉所指定的文件名不能已存在。

【说明】Name 语句重新命名文件并将其移动到一个不同的目录或文件夹中。如果有必要,Name 可跨驱动器移动文件。但当〈新文件名〉和〈旧文件名〉都在相同的驱动器中时,只能重新命名已经存在的目录或文件夹。

Name 语句不能用来创建新文件、目录或文件夹。

在一个已打开的文件上使用 Name 语句,将会产生错误。必须在改变名称之前,先关闭打开的文件。Name 参数中不能包括多字符(\*)和单字符(?)的通配符。

### 3. FileCopy 语句

FileCopy 语句用来复制一个文件。其语法格式为:

**FileCopy** 〈源文件〉,〈目标文件〉

其中,〈源文件〉是字符串表达式,用来表示要被复制的文件名,可以包含目录或文件夹及驱动器。〈目标文件〉也是字符串表达式,用来指定要复制的目标文件名,可以包含目录或文件夹及驱动器。

【说明】 如果想要对一个已打开的文件使用 FileCopy 语句,则会产生错误。

## 习题 14

### 一、选择题

14.1 VB 根据计算机访问文件的方式将文件分成 3 类,其中不包括( )。

A) 顺序文件              B) UNIX 文件              C) 二进制文件              D) 随机文件

14.2 下列说明中,不属于随机文件特点的是( )。

- A) 可以随意读取随机文件中任一记录的数据
- B) 随机文件没有只读或只写的操作方式,随机文件只要一打开,就既可读又可写
- C) 随机文件的操作是以记录为单位进行的
- D) 随机文件的读/写操作语句与顺序文件的读/写操作语句一样

14.3 以下叙述中正确的是( )。

- A) 一条记录中所包含的各个元素的数据类型必须相同
- B) 随机文件中每条记录的长度是固定的
- C) Open 命令的作用是打开一个已经存在的文件
- D) 使用 Input#语句可以从随机文件中读取数据

14.4 以下关于文件的叙述中,错误的是( )。

- A) 顺序文件中的记录是一个接一个顺序存放的
- B) 随机文件中记录的长度是随机的
- C) 执行打开文件的命令后,自动生成一个文件指针
- D) LOF 函数返回给文件分配的字节数

14.5 用 Write 和 Print 语句向文件中写入多个数据的差别在于 ( )。

- A) Write 语句不会自动在数据项之间插入逗号
- B) Print 语句自动在数据项之间插入逗号
- C) Write 语句写入字符串会自动给字符串加上双引号, 写入的正数前面没有空格
- D) Print 语句写入字符串会自动给字符串加上双引号, 写入的正数前面没有空格

14.6 设有语句

Open "C:\\Test.Dat" For Output As #1

则以下错误的叙述是 ( )。

- A) 该语句打开 C 盘根目录下下一个已存在的文件 Test.Dat
- B) 该语句在 C 盘根目录下建立一个名为 Test.Dat 的文件
- C) 该语句建立的文件的文件号为 1
- D) 执行该语句后, 就可以通过 Print#语句向文件 Test.Dat 中写入信息

14.7 要把一个记录型变量的内容写入文件中指定的位置, 所使用的语句格式为 ( )。

- A) Get 文件号, 记录号, 变量名
- B) Get 文件号, 变量名, 记录号
- C) Put 文件号, 变量名, 记录号
- D) Put 文件号, 记录号, 变量名

14.8 要获得当前驱动器, 应使用驱动器列表框的 ( ) 属性。

- A) Path
- B) Drive
- C) Dir
- D) Pattern

14.9 目录列表框的 Path 属性的作用是 ( )。

- A) 显示当前驱动器或指定驱动器上的路径
- B) 显示当前驱动器或指定驱动器中某目录下的文件名
- C) 显示根目录下的文件名
- D) 只显示当前路径下的文件

14.10 在窗体上画一个名称为 Drive1 的驱动器列表框, 一个名称为 Dir1 的目录列表框。当改变当前驱动器时, 目录列表框应该与之同步改变。设置两个控件同步的命令放在一个事件过程中, 这个事件过程是 ( )。

- A) Drive1\_Change
- B) Drive1\_Click
- C) Dir1\_Click
- D) Dir1\_Change

14.11 要使文件列表框中的文件随目录列表框中所选择的当前目录的不同而发生变化, 应该 ( )。

- A) 在 File1 中的 Change 事件中, 输入 File1.Path=Dir1.Path
- B) 在 Dir1 中的 Change 事件中, 输入 File1.Path=Dir1.Path
- C) 在 File1 中的 Change 事件中, 输入 Dir1.Path=File1.Path
- D) 在 Dir1 中的 Change 事件中, 输入 Dir1.Path=File1.Path

## 二、填空题

14.12 对随机文件数据存取是以\_\_\_\_为单位进行操作的。

14.13 文件根据数据性质, 可分为\_\_\_\_文件和\_\_\_\_文件。

14.14 下列程序的功能是: 将数据 1, 2, ..., 8 写入顺序文件 Num.txt, 请补充完整。

```
Private Sub Form_Click()
```

```
Dim i As Integer
```



```
Open "Num.txt" For Output As #1
For i=1 To 8
```

```
Next i
Close #1
```

End Sub

14.15 改变驱动器列表框的 Drive 属性值将引发\_\_\_\_\_事件。

三、编程题

14.16 在当前目录中有顺序文件 in7.txt（文件中只有字母和空格）。在窗体上画一个文本框，名称为 Text1，允许多行显示；再画 3 个命令按钮，名称分别为 C1, C2, C3，标题分别为“输入”、“转换”、“存盘”（如图 14-12 所示）。请编写适当的事件过程，使得在运行时，单击“输入”按钮，则从考生文件夹中读入 in7.txt 文件，放入 Text1 中；单击“转换”按钮，则把 Text1 中的所有小写字母转换为大写字母；单击“存盘”按钮，则把 Text1 中的内容存入 out7.txt 文件中。

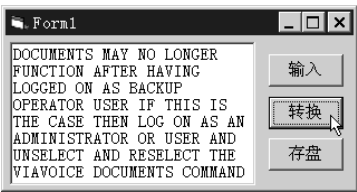


图 14-12 小写字母转换为大写字母

14.17 某单位全年每次报销的经费（假定为整数）存放在一个磁盘文件中，试编写一个程序，从该文件中读出每次报销的经费，计算其总和，并将结果存入另一个文件中。

14.18 编写一个程序，用来处理活期存款的结算事务。每次处理之后，程序都要显示当前的结存，并把它存入一个文件中，要求输出的浮点数保留小数点后两位。

14.19 编写程序，按下列格式输出月历，并把结果放入一个文件中。

SUN	MON	TUE	WED	THU	FRI	SAT
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

14.20 假定在磁盘上已建立了一个通信录文件，文件中的每条记录都包括编号、用户名、电话号码和地址 4 项内容。编写一个程序，用自己选择的检索方法从文件中查找指定用户的编号，并在文本框中输出其名字、电话号码和地址。

14.21 通过键盘输入数据，包括学号、姓名、性别及数学、英语、电子的成绩等数据，将这些数据输入到一个顺序文件（stu.dat）中。

14.22 通过键盘输入数据，包括学号、姓名、性别及数学、英语、电子的成绩等数据，将这些数据输入到一个随机文件（xsda2.dat）中。

14.23 从 stu.dat 中读入全部学生成绩数据，将其中获奖学金的学生数据存入一个新文件（stu1.dat）中。评奖的条件是：每门课程成绩均在 85 分以上或 3 门课程总分在 270 分以上。

14.24 编写应用程序，实现功能如下：

- （1）建立一个随机文件，管理某单位的职工情况，其中每条记录都由工作证号、姓名、

性别、工资、工作日期 5 项组成，可以向此文件添加新记录；

- (2) 可以修改、删除记录；
- (3) 可以按记录浏览所有职工的情况；
- (4) 可以按姓名查找，并显示找到的记录；
- (5) 可以按工作证号查找，并显示找到的记录。

14.25 设计一个用于登录或注册的对话框程序，程序启动后默认为登录状态，如图 14-13 所示。如果用户尚未获得合法的用户名和密码，可单击“注册”单选钮，屏幕显示隐藏的“确认密码”输入栏，如图 14-14 所示。

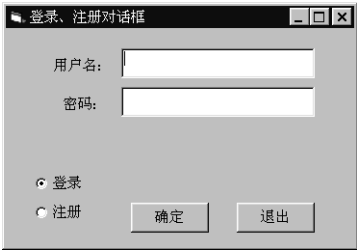


图 14-13 登录状态

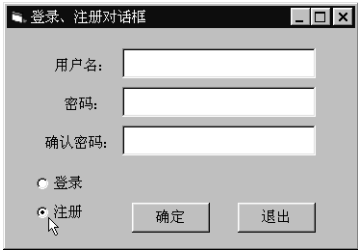


图 14-14 注册状态

14.26 利用随机文件读/写操作设计程序。在图 14-15 所示的程序界面中输入相关信息后，单击“追加记录”按钮，程序将计算出考生的总分，并显示添加成功的信息。如果在当前记录中输入希望查询的记录号后，单击“显示记录”按钮，则程序会将保存在数据文件中的记录显示在屏幕上，如图 14-16 所示。

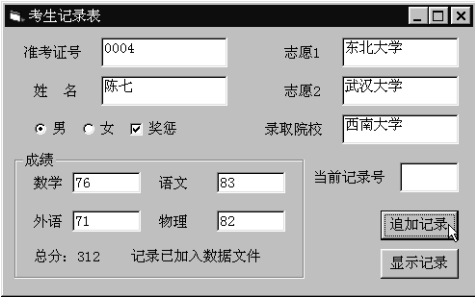


图 14-15 追加记录



图 14-16 显示指定记录

14.27 设计一个使用文件控件的示例程序，执行这个程序可以找到磁盘中保存的任何文件。在驱动器列表框中选择驱动器后，目录列表框和文件列表框中的内容自动发生相应的变化。选择目录列表框后，文件列表框中的内容自动发生相应的变化。双击某文件夹时，窗体上显示出当前路径。单击某文件时，在文本框中显示包括路径的文件名称。如果用户在文本框中输入正确的完整文件名（包括路径的文件名），并按回车键后，程序可以找到该文件；若找不到则显示错误信息。

## 第 15 章 数据库访问技术

VB 中提供了多种访问数据库的方法，可以访问的数据库类型有 dBase, FoxPro 和 Access 等本地数据库。另外，可以通过 ODBC 方式访问 MS SQL Server, Oracle SQL Server 和 Sybase SQL Server 数据库，并以客户-服务器方式存取数据库中的数据。

VB 提供的数据库访问方法主要有：使用数据（Data）控件或 ADO 数据控件访问数据库，通过 ODBC 方式访问远程数据库，以及采用对象变量访问数据库等。本章主要介绍最基本的使用数据控件和 ADO 控件访问本地数据库的方法。

### 15.1 数据库的概念

在使用数据库之前首先需要理解关于数据库的几个概念。

#### 1. 几个概念

数据库是信息时代的产物，是进行大量信息管理和处理的必需品。人们通过数据库可以方便地使用、查找需要的信息。

##### （1）数据库

所谓数据库（Database），是指一组排列得易于处理或读取的相关信息，它是由一个或多个表对象组成的集合。它类似于 Excel 的工作簿和工作表。

##### （2）数据库管理系统

数据库管理系统是指在操作系统支持下为数据库建立、使用和维护而配置的庞大软件，如 Microsoft SQL Server 和 Microsoft Access 等。它们建立在操作系统的基础上，对数据库进行统一的管理和控制。利用数据库管理系统提供的一系列命令，用户可以建立各种数据库操作文件和辅助文件，定义数据，以及对数据进行添加、删除、更新、查找、输出等操作。用户使用的各种数据库命令及应用程序的执行，都要通过数据库管理系统来实现。此外，数据库管理系统还承担着数据库维护的任务。

##### （3）数据库应用程序

数据库应用程序是指用 VB, FoxPro 等开发工具设计的、实现某种特定功能的应用程序，如学生成绩管理系统、工资管理系统、物资管理系统等。它利用数据库管理系统提供的各种手段访问一个或多个数据库，实现其特定的功能。

##### （4）数据库系统

数据库系统是由计算机硬件，操作系统，数据库管理系统，以及其他对象支持下建立起来的数据库、数据库应用程序，用户和维护人员等组成的一个整体。

#### 2. 关系型数据库

按数据组织形式可以将数据库分为层次型、网状型和关系型结构。其中最常用的是关系

型数据库。

关系型数据库由表、记录、字段组成。表的数据组织形式类似于一张二维关系表，每行称为一条记录，每列称为一个字段。一个数据库由若干张表组成，表与表之间通过关系来连接。

近几年来，关系型数据库已成为数据库设计事实上的标准，这不仅因为关系模型本身具有强大的功能，而且还由于它提供了称为结构化查询语言（Structure Query Language，简称 SQL）的标准接口，该接口允许以一致的和可以理解的方法来一起使用多种数据库工具和产品。

## 15.2 Access 数据库

Access 数据库管理系统是 Microsoft Office 的一个组件，是最常用的本地数据库之一。在 VB 中可以方便地使用数据控件和 ADO 控件来操作 Access 数据库。

### 15.2.1 创建 Access 数据库和表

#### 1. 创建 Access 数据库

执行“开始”→“程序”→“Microsoft Access”菜单命令，启动 Access。在图 15-1 所示的对话框中选择“空 Access 数据库”项，单击“确定”按钮。在图 15-2 所示的对话框中输入数据库文件名，并选择保存的位置后，单击“创建”按钮。

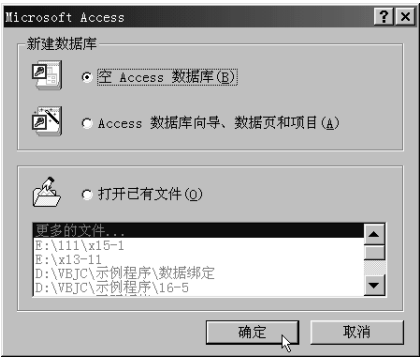


图 15-1 新建数据库对话框

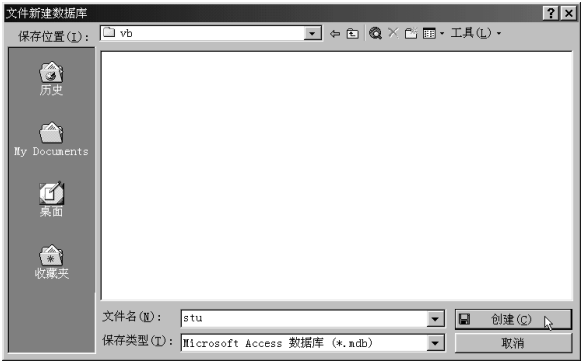


图 15-2 输入数据库文件名及保存位置

至此，一个空 Access 数据库创建完毕，并以指定的文件名（stu.mdb）保存在指定的文件夹中。用户可以继续创建需要的表，也可以退出 Access，待以后需要时将其打开，完成后续工作。

#### 2. 创建 Access 数据表

新建或打开数据库后，在图 15-3 所示的数据库对话框中，用户可以选择使用设计器、使用向导或通过输入数据的方法创建表。这里只介绍使用设计器创建表的方法。

双击“使用设计器创建表”项打开图 15-4 所示的创建表结构对话框，在此可以依次输入各字段的名称和数据类型。在“字段属性”区中输入字段的大小、格式等属性值。图中“常规”标签下的“有效性规则”项用来指定该字段能够接收的数据的准则，“有效性文本”项

是当出现违反“有效性规则”数据时显示在屏幕上的提示内容。

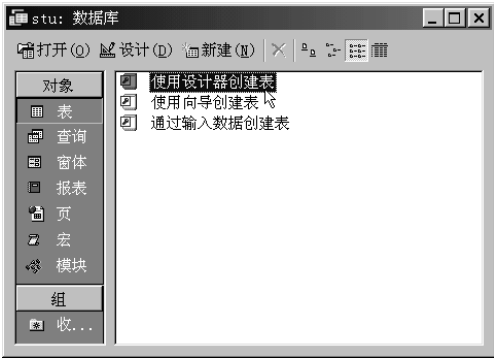



图 15-3 使用设计器创建表



图 15-4 创建表结构

一般在每个表中均应指定一个字段为该表的主键（如本例的“学号”字段），主键应唯一地代表一条记录，即所有记录中该字段没有重复的值。有了主键可以方便地与数据库中其他表进行关联，并利用主键值相等的规则结合多个表中的数据创建查询。

输入完毕后关闭创建表结构对话框，系统会提示为新建的表命名，此时新建的表将出现在数据库窗口中。如果需要修改表结构，可以在数据库窗口中选择了表名称后，单击工具栏中的“设计”按钮, 重新进入创建表结构对话框进行必要的修改。双击表名称可以打开如图 15-5 所示的表数据输入窗口，依次将各种数据输入到数据表中。需要注意的是，表中主键字段的值不允许空缺。输入完毕后关闭输入窗口，将数据保存在数据库文件中。

基本情况: 表							
学号	姓名	性别	年龄	家庭住址	联系电话	政治面貌	
0001	张三	女	18	河南郑州	0371-7654321	团员	
0002	李四	男	19	河南开封	0378-8765432	团员	
0003	王五	男	20	河北石家庄		党员	
0004	赵六	女	19	北京	010-8877665		
0005	陈七	男	18	山东济南		团员	
*			0				
记录: 5 共有记录数: 5							

图 15-5 输入表中各字段的数据

15.2.2 创建查询

如果在数据库中存在有多张数据表，并且各表中存在有相同的字段信息，此时为了避免数据冗余，通常利用数据表中的关系来减少表中的字段，如图 15-6 所示的学生成绩表中就省略了“姓名”、“性别”、“年龄”等字段。当需要使用综合信息时可以通过创建查询实现对多表的信息组合。

创建查询的步骤如下。

（1）单击数据库窗口中的“查询”按钮，双击“在设计视图中创建查询”项，如图 15-7 所示。

（2）打开查询设计器，如图 15-8 所示。在“显示表”对话框中，选择需要的表后单击“添

成绩: 表				
学号	计算机	数字电路	高等数学	
0001	87	78	64	
0002	88	81	92	
0003	78	78	64	
0004	64	82	88	
0005	70	81	98	
*	0	0	0	
记录: 5 共有记录数: 5				

图 15-6 学生成绩表

加”按钮将其添加到查询中（本例将“成绩”表和“基本情况”表添加到了查询中）。

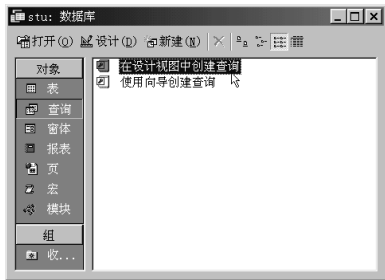


图 15-7 进入查询设计窗口

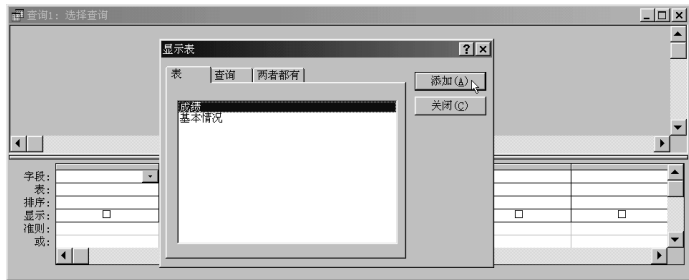


图 15-8 查询设计器

（3）由于在前面将“学号”字段定义成了主键，此时将自动建立一个“一对一”的关系（用一条连线表示）。将各表中需要的字段依次拖到查询字段列表区中，构成查询的字段框架，如图 15-9 所示。



图 15-9 创建查询字段

（4）如果需要向查询中添加一些计算字段，如“总分”、“平均分”等，可以使用字段生成器。在需要的位置上（如本例的“高等数学”之后）单击鼠标右键，从快捷菜单中选择“生成器”命令，打开图 15-10 所示的表达式生成器。输入表达式字段名后，用“:”分割后面的表达式内容。选择“成绩”表后，在字段列表区中双击将需要的字段添加到表达式中，单击工具栏中对应的符号按钮，生成表达式的计算式。输入完毕后，单击“确定”按钮。如果希望在“平均分”字段中设置保留小数点的位数，可以在“平均分”字段中单击鼠标右键，从弹出的快捷菜单中选择“属性”命令，在“格式”栏中输入“0.0”表示保留 1 位小数，如图 15-11 所示。



图 15-10 表达式生成器

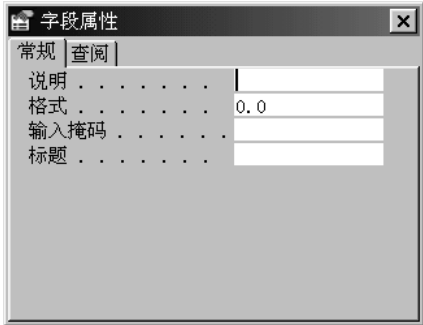


图 15-11 设置表达式的格式

设计完成关闭设计器窗口时，系统会提示输入新建查询的名称，并将其显示到数据库窗口中。双击查询名称可以看到查询中包含的数据，如图 15-12 所示。

需要说明的是，查询并没有新建任何数据的副本，只是将不同数据表的字段进行了重新组合，这些数据依然存放在原数据表中。

学号	姓名	性别	计算机	数字电路	高等数学	总分	平均分
0001	张三	女	87	78	64	229	76.3
0002	李四	男	88	81	92	261	87.0
0003	王五	男	78	78	64	220	73.3
0004	赵六	女	64	82	88	234	78.0
0005	陈七	男	70	81	98	249	83.0

图 15-12 新建查询的内容

## 15.3 使用数据控件

VB 通过使用数据控件、数据绑定控件（如文本框、组合框等标准控件）、数据访问对象、远程数据控件、ADO 数据控件来实现对数据库的访问。在这些工具中，数据控件和数据绑定控件是初学者最常用的工具，它们具有快捷、方便及功能强大等特点，甚至不需要编写任何程序代码，仅通过设置几个关键属性，使用一些类似于文本框这样的数据绑定控件，就可以实现对数据库的一般访问。

**【例 15-1】** 数据控件和数据绑定控件的使用方法示例。

新建一个工程，在窗体上添加两个文本框，两个标签和一个数据控件。

将标签的 Caption 属性分别设为“学号”和“姓名”。将数据控件 Data1 的 DatabaseName 属性设为具体的数据库文件名，如“d:\vb\stu.mdb”；RecordSource 属性设为数据库中具体的表或查询，如“基本情况”表。将 Text1 和 Text2 的 DataSource 属性设为 Data1（与数据控件 Data1 绑定），DataField 属性分别设为“学号”和“姓名”（绑定到具体的字段）。

程序启动后，绑定到字段的数据控件中将自动显示第一条记录的信息，单击数据控件的相应按钮即可在数据控件中浏览数据库中的内容，如图 15-13 所示。



图 15-13 数据控件和数据绑定控件的使用示例

### 15.3.1 数据控件的属性

数据控件是 VB 的标准控件之一，可以直接从工具箱中加入窗体。在数据控件的属性中，3 个基本属性（Connect, DatabaseName 和 RecordSource）决定了所要访问的数据资源。数据控件的常用属性如下。

#### 1. Connect（连接）属性

该属性用于定义所要连接的数据库类型，如“Access;”表示连接 Access 97 格式的数据库，“Access 2000;”表示连接 Access 2000 格式的数据库。需要说明的是，如果要在 VB 6.0

中使用 Access 2000 数据库，必须从 Microsoft 网站上下载并安装 VsSp5 补丁程序，否则在 Connect 属性取值列表中不会出现“Access 2000;”选项。

2. DatabaseName（数据库名）属性

该属性决定数据控件连接到哪个数据库上。对于多表数据库（如 Access 等），该属性为具体的数据库文件名，如：Data1.DatabaseName="d:\students.mdb"。对于单表数据库（如 FoxPro 等），该属性为数据库存放的目录，数据库文件名应存放在数据控件的 RecordSource 属性中。例如，需要访问 FoxPro 数据库文件“d:\fox\abcd.dbf”时应按如下方法设置属性：

```
Data1.DatabaseName="d:\fox"
Data1.RecordSource="abcd.dbf"
```

如果在设计或运行时，改变了数据控件的 DatabaseName 属性，应使用 Refresh（刷新）方法重新打开新数据库。

3. RecordSource（记录源）属性

该属性主要用来设置数据控件打开的数据库表名或查询名，它可以是一个表名、一个数据库中已存在的查询或一条 SQL 语句。如果在运行时通过代码改变了该属性值（连接到其他数据源），则必须使用 Refresh 方法使改变生效，并需要重建记录集（Recordset）。

4. BOFAction 和 EOFAction 属性

这两个属性用来指定当控件的 BOF 或 EOF 属性为 True（到达第一条记录之前或到达最后一条记录之后），而用户又单击了控件上的“◀”或“▶”按钮时，数据控件应该执行什么操作。取值情况分别见表 15-1 和表 15-2。

表 15-1 BOFAction 属性的取值

值	常 数	说 明
0	vbBOFActionMoveFirst	默认值，将记录指针指向第一条记录（将第一条记录作为当前记录）
1	vbBOFActionBOF	当 BOF 为 True 时触发数据控件的 Validate 事件，接着触发非法(BOF)记录上的 Reposition 事件。此时禁止数据控件上的“◀”按钮

表 15-2 EOFAction 属性的取值

值	常 数	说 明
0	vbEOFActionMoveLast	默认值，保持最后一条记录为当前记录
1	vbEOFAction	在 Recordset 的结尾移过去，并且在最后一条记录上触发数据控件的 Validate 事件，接着触发非法（EOF）记录上的 Reposition 事件。此时禁止数据控件上的“▶”按钮
2	vbEOFActionAddNew	移过最后一条记录，并且在当前记录上触发数据控件的 Validate 事件，然后自动触发 AddNew 事件和新记录上的 Reposition 事件

注意：当使用代码来操作数据控件创建的记录集对象时，EOFAction 属性是无效的，只有在使用鼠标操作数据控件的情况下它才有效。

5. ReadOnly 属性

该属性用来指定数据是否可以被编辑。如果 ReadOnly 属性设为 True，表示数据是只读



的。对数据控件来说，该属性只是在第一次打开数据库时才使用。若应用程序随后又打开了数据库的其他实例，则该属性被忽略。为了使此属性中的改变生效，首先要关闭数据库的所有实例，然后用 Refresh 方法刷新。

### 6. Exclusive 属性

该属性指定是否允许其他用户访问数据库。当其值为 True 时表示单用户使用，否则表示共享存取，即允许多用户访问数据库。

## 15.3.2 数据控件的事件

数据控件与其他 VB 控件一样支持许多事件，但除此之外数据控件还支持 Error, Reposition, Validate 等与数据库访问有关的事件。

### 1. Error 事件

该事件主要用来处理不能被任何应用程序捕获的错误。其语法格式为：

```
Private Sub Data1_Error(DataErr As Integer, Response As Integer)
.....    （错误处理过程）
End Sub
```

其中，DataErr 返回一个错误号。Response 的默认值为 1 时，表示显示错误信息；该值为 0 时，表示程序继续执行。

### 2. Reposition 事件

当用户单击数据控件上某个箭头按钮，或者在代码中使用了某个 Move 或 Find 方法使某条新记录成为当前记录时，将激发 Reposition 事件。

### 3. Validate 事件

在一条不同的记录成为当前记录之前，在 Update 方法之前（用 UpdateRecord 方法保存数据时除外）及 Delete, Unload 或 Close 操作之前会发生该事件。其语法格式如下：

```
Private Sub object_Validate (Action As Integer, Save As Integer)
    事件处理代码
End Sub
```

其中，Action 是一个整数，用来指示引发这种事件的操作。Save 是一个逻辑表达式，用来表示被连接的数据是否改变。

## 15.3.3 数据控件的方法

数据控件和其他控件一样也有自己的一些方法，常用的有：Refresh, UpdateRecord, UpdateControls 和 Close 方法。

### 1. Refresh 方法

该方法主要用来建立或重新显示与数据控件相连接的数据库记录集。若在程序运行时修改了数据控件的 DatabaseName, ReadOnly, Exclusive 或 Connect 属性，就必须使用该方法来刷

新记录集。该方法执行后，会将记录指针指向记录集中的第一条记录。

### 2. UpdateRecord 方法

通过该方法可以将数据绑定控件上的当前内容写入到数据库中，即可以在修改数据后调用该方法来确认修改。用这种方法在 `Validate` 事件期间将被连接的当前内容保存到数据库中，而不再激发 `Validate` 事件。

### 3. UpdateControls 方法

通过该方法可以将数据从数据库中重新读入到数据绑定控件中，即可以使用该方法放弃对数据绑定控件中数据的修改。

### 4. Close 方法

该方法主要用于关闭数据库或记录集，并且将该对象设置为空。

注意：在关闭数据库或记录集之前，必须使用 `Update` 方法更新数据库或记录集中的数据，以保证数据的正确性。

## 15.3.4 记录集对象

在 VB 中，数据库表是不能直接被访问的，VB 6.0 通过 Microsoft Jet 3.51 数据库引擎提供的记录集（`Recordset`）对象来检索和显示数据库记录。一个记录集对象表示一个或多个数据库表中的对象集合的多个对象，或运行一次查询所得到的记录结果。一个记录集对象相当于一个变量，与数据库表相似，记录集也是由行和列组成的，但不同的是记录集可以同时包含多个表中的数据。

VB 的 Jet 数据库引擎提供了大量的记录集属性和方法。通过引用数据控件的 `Database` 和 `Recordset` 属性，可以直接与数据控件一起使用这些属性和方法。

### 1. 记录集对象的属性

记录集对象的常用属性有：`BOF` 和 `EOF` 属性、`AbsolutePosition` 属性、`Bookmark` 属性及 `RecordCount` 属性。

#### （1）BOF 和 EOF 属性

这两个属性分别用来指示记录指针是否指向了第一条记录之前或最后一条记录之后。如果这两个属性同时为 `True`，则表示该记录集中无任何记录。

#### （2）AbsolutePosition 属性

该属性用于返回当前记录的序号，但不能将其作为记录编号的代替物，因为当执行了删除、添加、查询等操作后，记录的位置可能会改变。

#### （3）Bookmark 属性

该属性返回或设置当前记录集指针的书签，`Bookmark` 属性采用的是 `String` 类型。在程序中使用该属性重定位记录集的指针。下列语句使指针移到其他位置后迅速返回原位：

```
mybookmark=Data1.Recordset.BookMark      ' 设置书签保存当前记录指针位置
Data1.Recordset.MoveFirst                 ' 将记录指针移动到第一条记录
Data1.Recordset.BookMark=mybookmark      ' 使记录指针返回原位置
```

#### (4) RecordCount 属性

该属性是只读属性，用来获取记录集中的记录数。在多用户环境中，该属性返回的值可能是一个不准确的数，这与记录集对象被刷新的频率有关。为了获得准确的数据，在使用该属性前应先调用 MoveLast 方法。

## 2. 记录集对象的方法

记录集对象的常用方法有：AddNew, Edit, Delete, Move 和 Find 方法。

#### (1) AddNew 和 Edit 方法

使用 AddNew 方法可为数据库表添加一条记录。调用该方法将清除数据绑定控件中的所有内容，并且将一条空记录添加到记录集的末尾。

Edit 方法使当前记录集进入可以被修改状态。

新添加或修改后的记录，只有在执行了 Update 方法或通过数据控件移动了当前记录后，才会添加到数据库文件中。

#### (2) Delete 方法

该方法用来删除记录集中的当前记录。记录删除后，其内容仍显示在数据绑定控件中，应使用 Move 方法移动记录指针。删除记录时应先检查与该记录相关的关系后再删除，若数据库中存在某种必要的引用，则无法删除被引用记录。

#### (3) Move 方法

该方法用于记录指针的移动，常用于浏览数据库中的数据，包括以下 4 种方法。

- MoveFirst: 使记录集中的第一条记录成为当前记录。
- MoveLast: 使记录集中的最后一条记录成为当前记录。
- MoveNext: 下移一条记录，使下一条记录成为当前记录。
- MovePrevious: 上移一条记录，使上一条记录成为当前记录。

当一个记录集刚被打开时，第一条记录为当前记录。

#### (4) Find 方法

该方法用于在 Dynaset 和快照类型的记录集中查找符合指定条件的记录。若找到符合条件的记录，则将记录指针指向该记录，并将记录集对象的 NoMatch 属性设为 True；否则，将指针指向记录集的末尾，并将记录集对象的 NoMatch 属性设为 False。它包括以下 4 种方法。

- FindFirst: 查找符合条件的第一条记录。
- FindLast: 查找符合条件的最后一条记录。
- FindNext: 查找符合条件的下一条记录。
- FindPrevious: 查找符合条件的上一条记录。

例如，语句：

```
Data1.Recordset.FindFirst "姓名 like '李'"
```

用于查找姓名中包含“李”的第一条记录。

**【例 15-2】** 设计一个学生成绩管理程序，程序启动后显示数据库中记录总数、当前记录号及当前记录的各项数据，如图 15-14 所示。

用户可以在“学号”或“姓名”下拉列表框中，选择或输入内容后按回车键，查询指定学生的成绩，无此记录时显示提示信息，如图 15-15 所示。



图 15-14 程序主界面



图 15-15 未找到匹配记录

单击“添加”或“修改”按钮后，显示“输入口令”对话框，如图 15-16 所示。回答正确的口令后（本例为空字符串，可以直接单击“确定”按钮），显示一个空白的添加记录对话框。用户输入新记录的各项数据后，单击“更新”按钮使更新有效，并继续显示下一个添加记录对话框，单击“取消”按钮放弃数据，退出该对话框返回初始界面。

单击“修改”按钮后，“添加”和“修改”按钮变为“删除”和“更新”按钮，用户可以通过“学号”或“姓名”字段找到需要修改的记录。修改后单击“更新”按钮，将打开修改确认对话框，确认后完成当前记录的修改。选择其他记录继续进行修改工作，直到单击“退出”按钮返回初始界面（注意，在初始界面中单击“退出”按钮将退出程序）。

若单击“删除”按钮，将显示确认对话框，确认后当前记录将被删除。

设计步骤如下。

（1）创建数据库。

通过 Access 建立一个名为“成绩管理.mdb”的数据库，保存在 D 盘根目录下，在库中建立一个名为“成绩”的表，并向其中添加一些数据记录，表结构见表 15-3。

表 15-3 成绩表结构

字 段	类 型	大 小	字 段	类 型	大 小
学号	Text	3	数学	Single	4（系统自动设置）
姓名	Text	8	英语	Single	4
班级	Text	16（主界面未使用）	计算机	Single	4
总分	Single	4	平均分	Single	4

（2）设计程序界面。

本程序分为 3 个窗体，如图 15-17、图 15-18 和图 15-19 所示。启动窗体为“成绩管理”。在主窗体中添加 5 个按钮，其中“删除”和“更新”按钮启动时不可见，并且分别与“添加”、“修改”按钮重叠放置。其他控件的情况按图设置，这里不再赘述。



图 15-16 回答口令方可更改数据

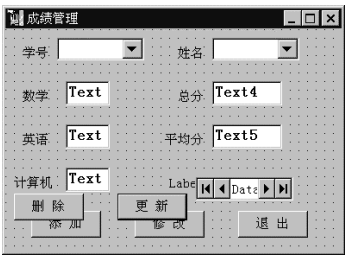


图 15-17 设计主程序界面

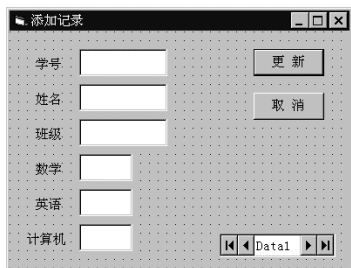


图 15-18 设计添加记录界面



图 15-19 设计输入口令界面

### (3) 设置对象的属性。

将“成绩管理”和“添加记录”窗体中的数据控件的 DatabaseName 属性设为数据库文件存放位置“d:\成绩管理.mdb”，RecordSource 属性设为“成绩”表。

将“成绩管理”窗体上两个 ComboBox 控件的 DataSource 属性设为 Data1（绑定到数据控件）。文本框的 Text 属性设为空，注意无须绑定到数据表字段。

### (4) 编写程序代码。

“成绩管理”窗体模块的代码如下：

```
Public panduan As Integer
' 定义全局变量判断用户单击的是“添加”还是“修改”按钮
' 在“输入口令”模块中还要用到该变量
Dim reccount As Integer ' 用来存放总记录条数
' 单击学号组合框时执行的程序代码

Private Sub Combo1_Click()
    Data1.Recordset.MoveFirst ' 将记录指针指向第一条记录
    Data1.Recordset.FindFirst "学号 = " & Combo1.Text & ""
    ' 上句指定了查询条件为“学号=组合框中显示的内容”，查找符合条件的第一条记录
    ' 注意书写格式，在引号中套用引号应使用单引号
    Combo2.Text = Data1.Recordset("姓名")
    Call xianshi
End Sub
' 在“学号”组合框中按回车键时执行的程序代码

Private Sub Combo1_KeyUp(KeyCode As Integer, Shift As Integer)
    If KeyCode = 13 Then
        Data1.Recordset.MoveFirst
        Data1.Recordset.FindFirst "学号 = " & Combo1.Text & ""
        Combo2.Text = Data1.Recordset("姓名")
        If Data1.Recordset.NoMatch Then ' 未找到匹配的记录，则显示提示信息
            MsgBox "查无此人！", 48, "注意"
        Else
            Call xianshi ' 调用 xianshi 过程
        End If
    End If
End Sub
```

End If

**End Sub**

**Private Sub Combo2\_Click()**

Data1.Recordset.MoveFirst

Data1.Recordset.FindFirst "姓名 = " & Combo2.Text & ""

Combo1.Text = Data1.Recordset("学号")

Call xianshi

**End Sub**

' 在“姓名”组合框中按回车键时执行的代码

**Private Sub Combo2\_KeyUp(KeyCode As Integer, Shift As Integer)**

If KeyCode = 13 Then

Data1.Recordset.MoveFirst

Data1.Recordset.FindFirst "姓名 = " & Combo2.Text & ""

Combo1.Text = Data1.Recordset("学号")

If Data1.Recordset.NoMatch Then

MsgBox "查无此人!", 48, "注意"

Else

Call xianshi ' 调用 xianshi 过程

End If

End If

**End Sub**

**Private Sub Command1\_Click()** ' 单击“添加”按钮时执行的代码

panduan = 1

Form2.Show 1 ' 显示输入口令对话框

**End Sub**

**Private Sub Command2\_Click()** ' 单击“修改”按钮时执行的代码

panduan = 2

Form2.Show 1

**End Sub**

**Sub gengxin()** ' 重新计算“总分”和“平均分”字段的值的自定义过程

Data1.Recordset.MoveFirst

Do While Data1.Recordset.EOF = False

Data1.Recordset.Edit ' 进入编辑状态

Data1.Recordset("总分") = Data1.Recordset("数学") + \_

Data1.Recordset("英语") + Data1.Recordset("计算机")

Data1.Recordset.Update ' 将缓冲区中的数据写入数据库

Data1.Recordset.Edit

Data1.Recordset("平均分") = Format(Data1.Recordset("总分") / 3, "0.0")

Data1.Recordset.Update

```

        Data1.Recordset.MoveNext
    Loop
End Sub
Sub xianshi()                                ' 刷新文本框中显示信息的自定义过程
    Text1 = Data1.Recordset("数学")
    Text2 = Data1.Recordset("英语")
    Text3 = Data1.Recordset("计算机")
    Text4 = Data1.Recordset("总分")
    Text5 = Format(Data1.Recordset("平均分"), "0.0") '保留 1 位小数
    Label8.Caption = "记录号: " & Data1.Recordset.AbsolutePosition + 1 & "/" & reccount
End Sub
Private Sub Command3_Click()                ' 单击“删除”按钮时执行的代码
    a = MsgBox("当前记录将被删除, 确定吗?", 4 + 48, "警告")
    If a = vbNo Then Exit Sub
    Data1.Recordset.Delete
    Data1.Refresh
    Combo1.Clear
    Combo2.Clear
    Call chushihua
End Sub
Private Sub Command4_Click()                ' 单击“更新”按钮时执行的代码
    a = MsgBox("当前记录将被修改, 确定吗?", 4 + 48, "警告")
    If a = vbNo Then Exit Sub
    Data1.Recordset.Edit
    With Data1
        .Recordset("学号") = Combo1.Text
        .Recordset("姓名") = Combo2.Text
        .Recordset("数学") = Text1
        .Recordset("英语") = Text2
        .Recordset("计算机") = Text3
        .Recordset("总分") = Val(Text1) + Val(Text2) + Val(Text3)
        .Recordset("平均分") = Format(.Recordset("总分") / 3, "0.0") ' 使用 Format 函数, 保留 1 位小数
    End With
    Combo1.Clear
    Combo2.Clear
    Data1.Refresh
    Call chushihua
    Call xianshi
End Sub

```

**Private Sub Command5\_Click()** ' 单击“退出”按钮时执行的代码

' 如果在“修改”状态下单击“退出”按钮，则返回初始界面；否则结束程序

If Command1.Visible = False Then

Command1.Visible = True

Command2.Visible = True

Command3.Visible = False

Command4.Visible = False

Else

End

End If

**End Sub**

**Private Sub Form\_Initialize()** ' 窗体初始化时执行的代码

Data1.Refresh

Call gengxin

Call chushihua

**End Sub**

**Sub chushihua()** ' 数据初始化自定义过程

Data1.Recordset.MoveFirst

Do While Data1.Recordset.EOF = False

Combo1.AddItem Data1.Recordset("学号") ' 将学号字段的内容添加至组合框列表

Combo2.AddItem Data1.Recordset("姓名") ' 将姓名字段的内容添加至组合框列表

Data1.Recordset.MoveNext

Loop

reccount = Data1.Recordset.RecordCount

Data1.Recordset.MoveFirst

Combo1.Text = Data1.Recordset("学号")

Combo2.Text = Data1.Recordset("姓名")

Call xianshi

**End Sub**

“输入口令”窗体模块的代码：

Dim cishu As Integer ' 用来存放输入口令的次数

**Private Sub Command1\_Click()** ' 单击“确定”按钮时执行的代码

If Text1 = "" Then ' 指定密码为一个空字符串

Unload Me

If Form1.panduan = 1 Then ' 用户单击了主窗体上的“添加”按钮

Unload Form1

Form3.Show 1

Else

Form1.Text1.Locked = False

Form1.Text2.Locked = False

Form1.Text3.Locked = False



```

        Form1.Command1.Visible = False
        Form1.Command2.Visible = False
        Form1.Command3.Visible = True
        Form1.Command4.Visible = True
    End If
Else
    If cishu < 2 Then                                ' 连续 3 次密码输入错误将退出本模块
        MsgBox "无效口令，请重新输入！", 48, "错误"
        Text1 = ""
        Text1.SetFocus
        cishu = cishu + 1
    Else
        MsgBox "你无权使用本功能！", 48, "警告"
        Unload Me
    End If
End If
End Sub
Private Sub Command2_Click()                    ' 单击“取消”按钮时执行的代码
    Unload Me
End Sub

```

“添加记录”窗体模块的代码：

```

Private Sub Command1_Click()                    ' 单击“更新”按钮时执行的代码
    If Text1 = "" Or Text2 = "" Or Text3 = "" Or Text4 = "" Or Text5 = "" Then
        MsgBox "请输入完整的数据！", 48            ' 若有空白项则显示提示信息，退出过程
        Text1.SetFocus
        Exit Sub
    End If
    With Data1
        .Recordset.AddNew
        .Recordset("学号") = Text1
        .Recordset("姓名") = Text2
        .Recordset("班级") = Text3
        .Recordset("数学") = Text4
        .Recordset("英语") = Text5
        .Recordset("计算机") = Text6
        .Recordset("总分") = Val(Text4) + Val(Text5) + Val(Text6)
        .Recordset("平均分") = Format(.Recordset("总分") / 3, "0.0")
        .Recordset.Update
    End With
    Text1 = "" : Text2 = "" : Text3 = "" : Text4 = ""
    Text5 = "" : Text6 = "" : Text1.SetFocus      ' 清除 6 个文本框中的内容，使 Text1 得到焦点

```

```
End Sub
Private Sub Command2_Click()           ' 单击“取消”按钮时执行的代码
    Unload Me
    Form1.Show
    Call Form1.chushihua
End Sub
```

## 15.4 使用 ADO 控件

ADO（ActiveX Data Object）数据访问接口，是美国微软公司提出的长期的数据访问策略，它实现了 RDO 的绝大多数功能，另外还增加了一些新的特征，它将逐步地取代 DAO（Data Access Object）和 RDO（Remote Data Object）成为主要的数据库访问接口。VB 6.0 可以很好地支持 ADO 和 OLE DB 数据访问模式。用户可以使用 ADO 快速建立数据库连接，并通过它方便地操作数据库。

### 15.4.1 ADO 数据控件的属性、方法和事件

ADO 数据控件和 VB 的内部控件数据控件很相似，用户可以利用其属性、方法和事件快速地创建与数据库的连接。

#### 1. ADO 数据控件与数据库相关的属性

##### （1）ConnectionString 属性

该属性是一个字符串，可以包含一个连接所需的所有设置值，在该字符串中所传递的参数是与驱动程序相关的。例如，ODBC 驱动程序允许该字符串包含驱动程序、提供者、默认的数据库、服务器、用户名称及密码等内容。该属性的参数说明见表 15-4。

表 15-4 ConnectionString 属性的参数说明

参 数	说 明
Provider	指定用于连接的数据源名称
File Name	指定基于数据源的文件名（如：一个永久性数据源对象）
Remote Provider	指定在打开一个客户端连接时使用的数据源名称
Remote Server	指定打开客户端连接时使用的服务器的路径与名称

##### （2）UserName 属性

当数据库受密码保护时，需要指定该属性。与 Provider 属性类似，这个属性可以在 ConnectionString 属性中指定。如果同时提供了一个 ConnectionString 属性及一个 UserName 属性，则 ConnectionString 属性中的值将覆盖 UserName 属性的值。

##### （3）Password 属性

Password 属性在访问一个受保护的数据库时是必需的。与 Provider 属性和 UserName 属性类似，如果在 ConnectionString 属性中指定了密码，则将覆盖在这个属性中指定的值。

##### （4）RecordSource 属性

该属性通常包含一个数据库表名、一个查询或一个存储过程调用，用于决定从数据库中

检索什么信息。

(5) Mode 属性

该属性决定想用记录集进行什么操作。例如，只是想要创建一个浏览界面，可以将该属性设为只读来获得性能的改善。

(6) CommandType 属性

该属性用于指定 RecordSource 属性的取值类型是一个表的名称、一个查询、一个存储过程，还是一个未知的类型，见表 15-5。

表 15-5 CommandType 属性的取值

值	常 数	说 明
8	adCmdUnknown	CommandType 属性中的命令类型未知（默认值）
1	adCmdText	为一条 SQL 语句
2	adCmdTable	为一个数据库表名
4	adCmdStoredProc	为一个存储过程

(7) BOFAction 和 EOFAction 属性

这两个属性用来指定当记录指针指向文件开始和末尾时的行为。提供的选择包括：停留在开始或末尾、移动到第一条或最后一条记录、在末尾添加一条新记录。

ADO 控件的属性一般可以通过控件的属性页进行设置。

**【例 15-3】** 使用 ADO 控件设计一个简单的数据库浏览程序。程序启动后界面如图 15-20 所示，用户可以通过单击窗体下方 ADO 控件的移动箭头改变文本框中显示的记录信息。

(1) 设计程序界面。

在窗体中添加 4 个标签、4 个文本框和一个 ADO 控件。ADO 是一个 ActiveX 控件，需要在“部件”对话框中，选择“Microsoft ADO Data Control 6.0”项，将其加入窗体。

(2) 设计对象属性。

将 4 个标签的 Caption 属性分别设为：“学号”、“姓名”、“总分”和“平均分”。4 个文本框的 Text 属性设为空。

鼠标指向窗体中的 ADO 控件，单击右键，在弹出的快捷菜单中选择“ADODC 属性”命令，将打开如图 15-21 所示的 ADO 控件“属性页”对话框。



图 15-20 程序启动后的界面



图 15-21 “属性页”对话框

在“通用”选项卡中选择“使用连接字符串”项后，单击“生成”按钮，打开如图 15-22 所示的“数据链接属性”对话框，在其中可以设置 ADO 控件的 ConnectionString 属性。在“数

据链接属性”对话框的“提供者”选项卡中，选择适当的 OLE DB 的提供者（本例选择了“Microsoft Jet 3.51 OLE DB Provider”项）后，单击“下一步”按钮，进入“数据链接属性”对话框的“连接”选项卡。在其中选择需要使用的数据库文件（本例选择了前面建立的“成绩管理.mdb”文件），然后单击“测试连接”按钮，验证连接的正确性。

在“数据链接属性”对话框的“高级”选项卡中可以设置访问权限。在“数据链接属性”对话框的“所有”选项卡中可以查看、编辑生成的连接字符串的所有内容，如图 15-23 所示。最后单击“确定”按钮完成 ConnectionString 属性的设置。



图 15-22 “数据链接属性”对话框



图 15-23 “所有”选项卡

在 ADO 控件“属性页”对话框的“记录源”选项卡中，设置 CommandType 为 2（表类型），设置控件的 RecordSource 属性为“成绩”表，如图 15-24 所示。在 ADO 控件“属性页”对话框的“身份验证”选项卡中可以设置控件的 Username 属性和 Password 属性，如图 15-25 所示。

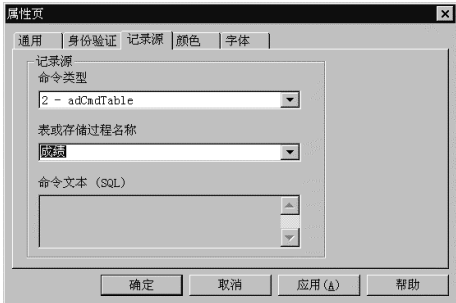


图 15-24 “记录源”选项卡

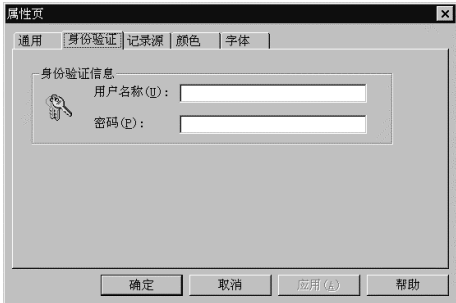


图 15-25 “身份验证”选项卡

在 ADO 控件的属性设置完毕后，设置 4 个文本框的 DataSource 属性为 ADO 控件（Adodc1），DataField 属性分别为“学号”、“姓名”、“总分”和“平均分”（分别绑定到相应的字段）。

（3）编写程序代码。

```
Private Sub Text1_Change()  
    a = Adodc1.Recordset.AbsolutePosition ' 当前记录号  
    b = Adodc1.Recordset.RecordCount     ' 数据源中记录的总数
```

```
Adodc1.Caption = "当前记录号: " & a & "/" & b
```

```
End Sub
```

## 2. ADO 记录集对象的方法

(1) UpdateControls 方法、UpdateRecord 方法、AddNew 方法、Delete 方法和 Move 方法这一组方法与前面介绍过的数据控件对应方法基本一致，此处不再赘述。

### (2) CancelUpdate 方法

取消添加、修改记录的操作，恢复到更改以前的状态。

### (3) UpdateBatch 方法

保存添加的记录或修改以后的内容。

## 3. ADO 数据控件的事件处理方法

### (1) WillMove 和 MoveComplete 方法

WillMove 方法在挂起操作更改记录集中的当前位置前调用，MoveComplete 方法则在记录集的当前位置更改完成时调用。

### (2) WillChangeField 和 FieldChangeComplete 方法

WillChangeField 方法在挂起操作对记录集中的一个或多个 Field 对象值进行更改前调用，FieldChangeComplete 方法在一个或多个 Field 对象值已经更改后调用。

### (3) WillChangeRecordset 和 RecordsetChangeComplete 方法

WillChangeRecordset 方法在挂起的操作更改记录集前调用，RecordsetChangeComplete 方法在记录集更改后调用。

## 15.4.2 高级数据绑定控件

### 1. DataGrid 控件

DataGrid（数据网格）控件是一种类似于电子表格的数据绑定 ActiveX 控件（Microsoft DataGrid Control 6.0），需要配合 ADO 控件一起使用。它用若干行、列来表示记录集对象的记录和字段。可以使用 DataGrid 控件创建一个允许用户阅读和写入绝大多数数据库的应用程序中。DataGrid 控件可以在设计时快速进行配置，只需少量代码或无须代码。当在设计时设置了 DataGrid 控件的 DataSource 属性后，就会用数据源的记录集来自动填充该控件，以及自动设置该控件的列标头。可以编辑该网格的列，删除、重新安排、添加列标头，或者调整任意一列的宽度。

在运行时，可以在程序中切换 DataSource 属性来查看不同的表，或修改当前数据库的查询，以返回一个不同的记录集合。

DataGrid 控件的大多数属性都可以通过其属性页进行设置，方法是：鼠标指向添加到窗体上的 DataGrid 控件，单击右键，在弹出的快捷菜单中选择“属性”命令，如图 15-26 所示，此时将打开如图 15-27 所示的 DataGrid 控件的“属性页”对话框。



图 15-26 执行快捷菜单中的“属性”命令



图 15-27 DataGrid 控件的“属性页”对话框

**【例 15-4】** 利用 DataGrid 控件设计一个具有数据库浏览、修改、添加、删除记录等功能的程序。使用前面已经建立的“成绩管理.mdb”数据库，要求：不显示“班级”字段，“总分”及“平均分”字段的内容不允许编辑，且数据能够自动计算。调整网格各列的宽度，使之能够在屏幕宽度内显示所有列的内容。

设计步骤如下。

(1) 设计程序界面。

在窗体上添加一个 ADO 数据控件和一个 DataGrid 控件。

(2) 设置对象属性。

参照前面介绍的方法，建立 ADO 控件与数据库文件“d:\成绩管理.mdb”的连接。设置 DataGrid 控件的 DataSource 为 Adodc1（连接到 ADO 控件），将 ADO 控件设为不可见。

在图 15-26 所示的快捷菜单中执行“检索字段”命令，出现“检索字段”对话框，询问是否要以新的字段定义替换现有网格布局，回答“是”，使数据源中所有字段名显示在数据网格的列标头中。而后在图 15-26 所示的快捷菜单中执行“编辑”命令。然后鼠标指向“班级”列单击鼠标右键，在弹出的快捷菜单中执行“删除”命令，如图 15-28 所示，使网格中不显示该字段（注意此时鼠标的外观样式及菜单内容）。

在编辑状态下，用鼠标拖动各字段列标题的交界线可以改变各列的宽度。调整窗体的宽度，使之与网格同宽。

在 DataGrid 控件的“属性页”对话框的“通用”选项卡中，选中“允许添加”和“允许删除”复选框。在“键盘”选项卡中选择“Tab 键动作”为 1（按 Tab 键时可以横向移动到下一单元格），选中“允许箭头”复选框（可以使用方向键移动单元格位置），如图 15-29 所示。



图 15-28 删除班级字段



图 15-29 设置 Tab 键动作

在“布局”选项卡中分别选择“总分”和“平均分”列，并选中“锁定”复选框，如图 15-30 所示。

在“格式”选项卡中，选择平均分列，并设置数据格式，如图 15-31 所示。

另外，在“列”选项卡中，可以设置某一字段显示在 DataGrid 中的列标题，这对数据源中采用英文字段名的情况十分有用。在“字体”选项卡中可以分别设置标题和数据的字体、字号及简单效果（如加粗、斜体等）。在“颜色”选项卡中，可以设置前景色和背景色。在“拆分”选项卡中，可以设置由图 15-30 所示的“拆分”命令建立的各个网格块的属性。

(3) 编写程序代码如下。

Private Sub Form\_Resize()

DataGrid1.Align = 1

DataGrid1.Height = Form1.ScaleHeight

' 窗体大小变化时执行的代码

' 使数据网格顶端对齐到窗体

' 使数据网格的宽度自适应于窗体



图 15-30 设置布局

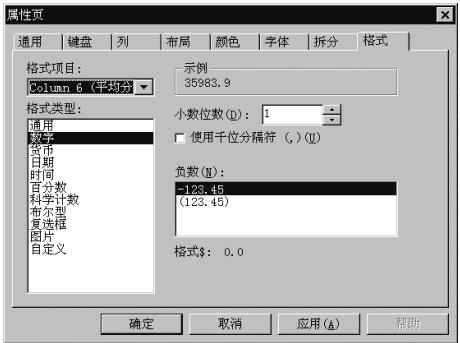


图 15-31 设置数据格式

Form1.Caption = "DataGrid 控件应用示例"

End Sub

' 网格中数据被编辑后执行的程序代码

Private Sub DataGrid1\_AfterColEdit(ByVal ColIndex As Integer)

' 若被编辑的是 2, 3, 4 列（计算机、英语或数学），则重新计算 5, 6 列（总分、平均分）的值

If ColIndex = 2 Or ColIndex = 3 Or ColIndex = 4 Then

DataGrid1.Columns(5) = Val(DataGrid1.Columns(2)) + Val(DataGrid1.Columns(3)) + \_  
Val(DataGrid1.Columns(4))

DataGrid1.Columns(6) = DataGrid1.Columns(5) / 3

End If

End Sub

Private Sub DataGrid1\_BeforeDelete(Cancel As Integer) ' 执行删除记录操作前执行的代码

' 记录被删除前显示警告信息，用户选择“否”，则取消删除操作

a = MsgBox("记录将被删除，确定吗？", 48 + 4, "警告")

If a = vbNo Then Cancel = True

End Sub

若光标已指向了最后一条记录，用户又按了下移光标键或者直接单击图 15-32 所示的最后空行中的某一单元格，则在结尾处自动添加一条记录，并进入编辑状态。输入具体数据后

按回车键或将光标移至他处，数据将被写入数据库中。

如图 15-33 所示，在某条记录的最左端单击可以选中整条记录（注意此时鼠标的外观），按 Delete 键可以删除选中的记录。

在上例中主要使用了 DataGrid 控件的“属性页”对话框进行控件属性的设置，也可以在程序中用代码进行同样的设置，详细情况请参阅 MSDN 帮助系统。



学号	姓名	计算机	英语	数学	总分	平均分
006	006	88	100	88	276	92.0
007	007	58	74	81	213	71.0
008	008	54	87	94	235	78.3
009	009	82	87	91	260	86.7
010	010	78	74	81	233	77.7
011	011	67	91	62	220	73.3
013	013	74	84	85	243	81.0
014	014	78	78	78	234	78.0
015	015	64	78	90	232	77.3
016	016	67	68	92	227	75.7
017	017	83	81	64	228	76.0

图 15-32 DataGrid 控件应用示例



学号	姓名	计算机	英语	数学	总分	平均分
007	007	58	74	81	213	71.0
008	008	54	87	94	235	78.3
009	009	82	87	91	260	86.7
010	010	78	74	81	233	77.7
011	011	67	91	62	220	73.3
013	013	74	84	85	243	81.0
014	014	78	78	78	234	78.0
015	015	64	78	90	232	77.3
016	016	67	68	92	227	75.7
017	61	86	78	85	219	73.0

图 15-33 选中记录

2. DataList 控件和 DataCombo 控件

DataList 和 DataCombo 控件是类似于 ListBox 和 ComboBox 控件的 ActiveX 数据绑定控件（Microsoft DataList Control 6.0），这两个控件常用于如下两个方面。

（1）在关系型数据库中，可用这两个控件将数据从一个表输入到另一个表中。例如，在一个商品管理数据库中，每种商品的生产厂商和产品编号存放在一个表中，商品的库存数量及包括编号在内的其他情况存放在另一个表中。在提供产品编号给商品表时，利用 DataList 控件可以显示厂商的名称。

（2）通过下拉列表中选择或输入条件，可缩小搜索范围。DataList 和 DataCombo 控件的作用及属性基本相同，都是直接从 ADO 控件中得到信息的（这一点与 ListBox 和 ComboBox 控件需要使用 AddItem 方法不同），只是 DataCombo 控件支持用户的输入操作。

DataList 和 DataCombo 控件的主要属性，参见表 15-6。

表 15-6 DataList 和 DataCombo 的主要属性

属 性	说 明
RowSource	设置一个指定数据控件的值，DataList 控件和 DataCombo 控件的列表由这个数据控件填充，运行时不可用
ListField	返回或设置记录集对象中的字段名，这个对象由 RowSource 属性指定，用于填充 DataCombo 控件或 DataList 控件的列表部分
BoundColumn	返回或设置一个记录集对象的源字段的名称，该记录集对象用来为另一个记录集对象提供数据值
DataSource	返回或设置一个数据源，通过该数据源，数据使用者被绑定到一个数据库
SelectedItem	返回一个值，包含 DataCombo 控件或 DataList 控件中选中的记录的书签
MatchEntry	返回或设置一个值，它指示 DataCombo 控件或 DataList 控件如何在用户输入的基础上执行查找

如果需要使用单个 ADO 控件的 DataList 或 DataCombo 控件，应将 DataSource 属性与 RowSource 属性设置为同一个 ADO 控件，并将 DataField 与 BoundColumn 属性设为该 ADO 控件记录集中的相同字段。在这种情况下，列表就会用已更新的同一记录集中的 ListField 属性来填充。如果指定了 ListField 属性，却没有指定 BoundColumn 属性，则 BoundColumn 属性会自动设置为 ListField 属性。

【例 15-5】 利用 DataCombo 控件设计一个能够按班级进行成绩查询的程序。用户从



DataCombo 控件中选择某一班级名称,或直接输入班级名称并按回车键后,在下面的 DataGrid 控件中显示相应的内容,如图 15-34 所示。

(1) 建立数据库。

利用 Access 或 VisData 建立一个名为“学生成绩.mdb”的数据库。在数据库中建立两张数据表“成绩”和“班级”,其结构分别见表 15-7 和表 15-8。

表 15-7 “成绩”数据表结构

字段名	学号	姓名	班级代码	计算机	数字电路	高等数学	总分	平均分
类型	Text	Text	Text	Single	Single	Single	Single	Single
大小	6	8	5	4	4	4	4	4

表 15-8 “班级”数据表结构

字段名	班级代码	班级名称
类型	Text	Text
大小	5	10

数据表建立后,录入一些模拟数据。成绩表中的“总分”和“平均分”字段值可以通过 SQL 语句在数据库中计算,本程序由于篇幅所限不具有计算、添加、修改等功能,有兴趣的读者也可以自行对本程序进行修改。

(2) 设计程序界面。

在窗体上添加两个 ADO 控件,一个 DataCombo 控件,一个 DataGrid 控件和一个标签,如图 15-35 所示。

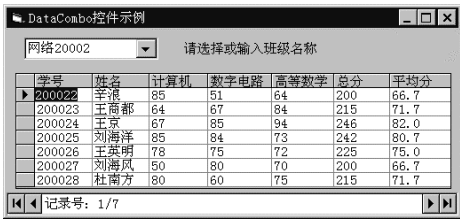


图 15-34 DataCombo 控件示例



图 15-35 设计程序界面

(3) 设置对象属性。

将 Adodc1 连接到“成绩”数据表,在“数据源”选项中设置命令类型为 2 (表类型),数据表选择“班级”表。Adodc2 连接到“班级”数据表,设置命令类型为 8 (未知类型),命令文本设置为一条 SQL 语句“Select \* From 成绩”,设置 Adodc2 的 Align 属性为 2(底端对齐)。

设置 DataCombo 控件的 RowSource 属性为 Adodc1, ListField 属性为“班级名称”。

按例 15-4 介绍的方法在 DataGrid 控件中屏蔽“班级代码”字段的显示,调整各列的宽度,锁定所有字段(只允许查询),设置“总分”及“平均分”字段的数据格式为保留一位小数。

(4) 编写程序代码。

```
Private Sub DataCombo1_Click(Area As Integer) '在 DataCombo 中发生单击事件时执行的代码
    If Area = 2 Then '用户单击选项区,而非箭头或文本输入区
        Call chaxun
        DataGrid1.SetFocus '数据网格得到焦点使用户可以用方向键查询记录
    End If
End Sub
'用户在 DataCombo 中按回车键时执行的代码
Private Sub DataCombo1_KeyPress(KeyAscii As Integer)
    If KeyAscii = 13 Then
```

```

        Call chaxun
    End If
End Sub
Sub chaxun()          ' 自定义查询过程
    a = DataCombo1.BoundsText
    Adodc1.Recordset.MoveFirst
    Adodc1.Recordset.Find ("班级名称=" & a & "")
    ' 以上代码使程序从当前记录中得到所选班级的班级代码
    If Adodc1.Recordset.EOF Then
        MsgBox "未查到指定班级", 48, "注意"
        Exit Sub
    End If
    b = "select * from 成绩 where 班级代码=" & Adodc1.Recordset("班级代码") & ""
    ' SQL 语句，用于返回班级代码为指定值的所有记录
    Adodc2.RecordSource = b
    Adodc2.Refresh
    Adodc2.Caption = "记录号: " & Adodc2.Recordset.AbsolutePosition & "/" & _
    Adodc2.Recordset.RecordCount
End Sub
Private Sub Form_Activate() ' 窗体激活时执行的代码
    ' 在 DataCombo 中显示当前记录的班级名称
    DataCombo1.BoundsText = Adodc1.Recordset("班级名称")
    ' 使数据网格得到焦点
    DataGrid1.SetFocus          ' 不执行该语句会使程序启动后直接显示成绩表中的所有记录
    Call chaxun
    Adodc1.Visible = False      ' 使 Adodc1 控件不可见
    Label1.Caption = "请选择或输入班级名称"
    Form1.Caption = "DataCombo 控件示例"
End Sub

```

### 15.4.3 使用数据窗体向导

VB 提供的数据库窗体向导可以帮助用户快速建立一般化的数据库应用程序，它可以根据用户的选择自动设置前面介绍过的 ADO 控件和数据绑定控件。

数据窗体向导是作为外接程序存在的，因此当一个新工程启动时，它并没有出现在系统菜单中。在使用之前应执行“外接程序”→“外接程序管理器”菜单命令，在打开的对话框中，选择“数据窗体向导”项并选择加载方式后单击“确定”按钮，将其加入到系统菜单中。

如果数据窗体仅是程序的一部分，也可以通过执行“工程”→“添加窗体”菜单命令，在打开的对话框中选择“数据窗体向导”项来启动该向导。

使用数据窗体向导的步骤如下。

(1) 从“外接程序”菜单或“添加窗体”对话框中启动数据窗体向导。若以前使用过

该向导，并保存了配置文件，在图 15-36 所示的“数据窗体向导-介绍”对话框中可以装载原来的设置，并单击“下一步”按钮。

(2) 在打开的“数据窗体向导-数据库类型”对话框中，选择本地数据库 Access 或远程数据库 Remote(ODBC)的数据库类型，选择完毕后单击“下一步”按钮。

(3) 根据上面的选择，将打开不同的对话框。图 15-37 所示为 Access 的“数据窗体向导-数据库”对话框，图 15-38 所示为 Remote(ODBC)的“数据窗体向导-连接信息”对话框。本例选择了本地硬盘上的 Access 数据库“成绩管理.mdb”。单击“下一步”按钮。



图 15-36 “数据窗体向导 - 介绍”对话框



图 15-37 “数据窗体向导 - 数据库”对话框

(4) 在打开的“数据窗体向导-Form”对话框中，用户可以指定窗体名称、窗体布局样式和绑定类型。在该对话框中，VB 为用户提供了 5 种窗体布局方式：单个记录、网格、主表/细表、MS HFlexGrid（分层表格）和 MS Chart（图表）。当选择了某种样式后，在预览区可以看到大体的外观，如图 15-39 所示。



图 15-38 “数据窗体向导 - 连接信息”对话框



图 15-39 “数据窗体向导- Form”对话框

在该对话框中 VB 还为用户提供了 3 种绑定类型。

- ADO 数据控件：使用 ADO 数据控件和数据绑定控件。
- ADO 代码：不使用任何 ADO 控件，由数据窗体向导自动生成所有操作控件的代码（使用 ADO 对象）。
- 类：创建一个提供数据访问功能的类模块。

本例选择了单个记录、ADO 数据控件，窗体名为 frmChengji（代码中要用到），单击“下一步”按钮。

(5) 在打开的“数据窗体向导-数据源”对话框中，用户需要指定某一表为数据源，并从可用字段中选择需要显示在窗体中的字段，调整显示字段的排列顺序，设置排序依据，设置完毕后单击“下一步”按钮。

(6) 在打开的“数据窗体向导-控件选择”对话框中，用户可以选择窗体中出现的可选

按钮控件（删除、添加、刷新等）。单击“下一步”按钮。

（7）在“数据窗体向导-已完成”对话框中，用户可以将以上各项设置保存成配置文件以备今后继续使用。最后单击“完成”按钮，VB 将自动生成程序界面及主要程序代码，如图 15-40 和图 15-41 所示。

至此，一个具有基本数据库管理功能的应用程序设计完毕。用户可以在此基础上对代码或窗体控件进行修改，以增强程序的功能。若希望程序启动时直接显示数据窗体，应在“工程”菜单中设置“工程属性”，将数据窗体指定为启动窗体。



图 15-40 自动生成的程序界面

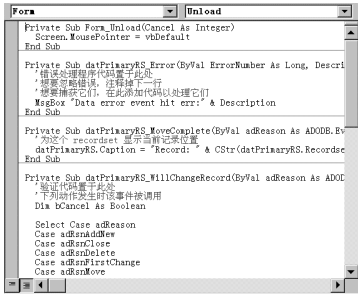


图 15-41 自动生成的程序代码

## 习题 15

### 一、选择题

- 15.1 下述关于数据库系统的叙述中正确的是（ ）。
- A) 数据库系统减少了数据冗余      C) 数据库系统中数据的一致性是指数据类型一致
- B) 数据库系统避免了一切冗余      D) 数据库系统比文件系统能管理更多的数据
- 15.2 下列有关数据库的描述，正确的是（ ）。
- A) 数据处理是将信息转化为数据的过程
- B) 数据的物理独立性是指当数据的逻辑结构改变时，数据的存储结构不变
- C) 关系中的每一列称为元组，一个元组就是一个字段
- D) 如果一个关系中的属性或属性组并非该关系的关键字，但它是另一个关系的关键字，则称其为本关系的外关键字
- 15.3 应用数据库的主要目的是（ ）。
- A) 解决数据保密问题      B) 解决数据完整性问题
- C) 解决数据共享问题      D) 解决数据量大的问题

### 二、填空题

- 15.4 数据库系统的核心是\_\_\_\_\_。
- 15.5 数据库设计包括两个方面的设计内容，它们是\_\_\_\_\_和\_\_\_\_\_。

### 三、编程题

15.6 设某校规定，超过全班平均成绩 10%者可以享受一等奖学金，超过全班平均成绩 5%者可以享受二等奖学金。试编写一个程序，使用数据控件，建立与存放学生成绩的 Access

数据库的连接，设数据库中包括“学号”、“姓名”、“成绩”3个字段。程序执行后，输出奖学金等级一览表。

15.7 使用数据控件，结合 Access 数据库设计一个“通信录”程序。程序启动后显示图 15-42 所示的界面，用户可以使用“姓名”和“地址”下拉列表框查询需要的记录。单击“更新”按钮时，显示图 15-43 所示的验证口令对话框。



图 15-42 程序启动时的界面



图 15-43 验证更新权限口令

用户在正确回答了口令之后，显示网格格式更新、删除、添加记录窗体。修改完毕后单击“更新”按钮将新数据写入数据库，输入新的数据可以添加记录，单击“删除”按钮，将删除当前记录，将光标移到最后的空白记录处。

15.8 使用 DBGrid 控件和数据绑定技术，创建一个数据库浏览程序。要求使用 Access 创建一个数据库，该数据库中包含有一个“学生成绩”表，表中有“学号”、“姓名”、“班级”、“年龄”、“性别”、“数学”、“语文”、“英语”和“计算机”字段。

15.9 使用 ADO 控件设计一个数据库管理程序，使之具有浏览、更新、添加和删除记录的功能。程序启动时，界面如图 15-44 所示，单击 ADO 控件的对应按钮，可以浏览数据表中的内容。单击“添加”按钮，出现图 15-45 所示的空记录界面，输入信息后在最后一个“性别”文本框中按 Enter 键可继续添加新记录，直到用户按下“更新”按钮。



图 15-44 程序启动时的界面

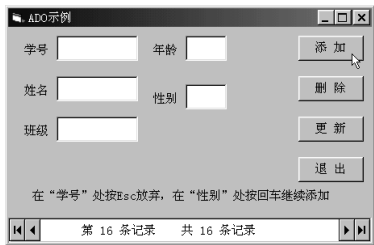


图 15-45 添加记录

单击“删除”按钮程序显示警告，确认后将删除当前记录，如图 15-46 所示。用户修改或添加了记录之后，单击“更新”按钮时才会将最后的更新一次性地写入数据表中，如图 15-47 所示。



图 15-46 删除记录

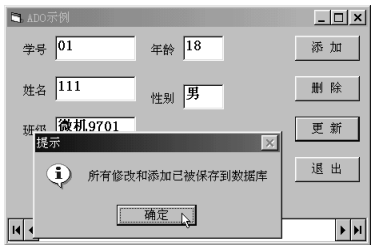


图 15-47 更新记录